$\langle M, \eta, \star \rangle$ Monads for Conventional Implicatures[*]

Gianluca Giorgolo
Ash Asudeh
Carleton University & University of Oxford

1 Introduction

The heteregenous group of expressions that contribute to conversation by adding slightly off-topic information has recently received a lot of attention from the semantics community, especially since the publication of Potts (2005). This is probably because, in the words of Barker et al. (2010), these expressions "challenge traditional conceptions of compositionality", but also because they seem to live on both sides of the semantics-pragmatics divide. In this paper we propose an analysis of these expressions that tries to reconcile them with compositionality and at the same time sheds some light on the central issue of the relation between semantics and pragmatics.

The reason why these expressions are problematic for standard approaches to compositionality is well illustrated by the following two examples:

(1) A: Most fucking neighbourhood dogs crap on my damn lawn.
B: No, that's not true.
$\Rightarrow$ No, the neighbourhood dogs don't crap on your lawn.
$\not\Rightarrow$ No, there's nothing wrong with dogs and/or their crapping on your lawn.

(2) A: John Lee Hooker, the bluesman from Tennessee, appeared in *The Blues Brothers*.
B: No, that's not true.
$\Rightarrow$ No, John Lee Hooker did not appear in *The Blues Brothers*.
$\not\Rightarrow$ No, John Lee Hooker was not from Tennessee.
B: True, but actually John Lee Hooker was born in Mississipi

The sentence uttered by A in (1) seems to convey the information that the majority of the dogs in the neighbourhood defecate on the lawn of the speaker together with the fact that A has a generally negative attitude towards the neighbourhood dogs (or just the defecating ones) and/or

[*]This research is supported by an Early Researcher Award from the Ontario Ministry of Research and Innovation and NSERC Discovery Grant #371969. The authors thank Avery Andrews, Chris Barker, Luca Ducceschi, Patrick Grosz, Daniel Gutzmann, Chung-chieh Shan and the audience at Sinn und Bedeutung 16 for their comments and questions.

their defecatory habits. The information about A’s attitude is evidently conveyed by the two expressives *fucking* and *damn*. However the reply of B seems to target only the first piece of information. B seems to challenge only the fact that the neighbourhood dogs defecate on A’s lawn, not the attitude of A towards the dogs.

Similarly in (2) the information conveyed by the appositive *the bluesman from Tennessee* cannot be negated by B by replying with *No, that’s not true*, as that would target only the proposition that John Lee Hooker appeared in the *The Blues Brothers*. Instead B would have to resort to a different conversational strategy, first agreeing with A that John Lee Hooker appeared in the movie, but then adding that the information about his birth place is wrong.

On the basis of cases like those above and other similar considerations, Potts (2005, 2007) has proposed that expressions such as non-restrictive relative clauses, parentheticals, nominal appositives and expressives require the postulation of more than one level of semantic content. Potts calls these levels *dimensions* and identifies, for the class of expressions under discussion (at least), two distinct dimensions:

1. an ‘at-issue’ dimension, which represents the aspect of meaning that is under discussion and is sensitive to logical operators such as negation; in the examples above the content of the main clauses (that is excluding the content of the expressive and the nominal appositive) contributes to the at-issue dimension and is the target of the negative replies by B,
2. a ‘side-issue’ dimension, also known as the ‘Conventional Implicature dimension’, represents an aspect of meaning that contributes information that is speaker-oriented, often peripheral, and not under discussion or up for grabs; the expressives content in (1) conveys clearly speaker-oriented information (A’s feelings towards the neighbourhood dogs) to which the interlocutor has limited access and therefore cannot easily discuss; in the case of the appositive in (2) the birth place of the musician is introduced as a kind of off-topic comment, not as part of the central discussion about his appearance in a movie.

However not all dimensions are born equal. In fact, a crucial aspect of Potts’s analysis is that the interactions between the at-issue and the CI dimensions are restricted in terms of flow of information. According to the theory set up by Potts, semantic content can flow from the at-issue dimension to the CI dimension but not *vice versa*. In other words, the interpretation of expressions contributing to the CI dimension can reuse semantic material introduced in the at-issue dimension but there are no lexical items that recycle material belonging to the CI dimension and introduce it in the at-issue one.

This last claim has been recently called into question by AnderBois et al. (2010) in light of examples like the following:

(3) John$_1$, who by the way almost destroyed his$_1$ car yesterday, has bought a motorcycle, too.

What we observe is a complex network of interactions between the two dimensions. The entity introduced by *John* and the associated discourse referent are respectively used as the anchor point for the relative pronoun heading the relative clause and as the antecedent for the anaphoric pronoun *his* as indicated by the indexes. This type of interaction is expected in Potts’s theory. However, we also have unexpected information flow from the side-issue content to the at-issue content, since the presupposition triggered by *too* is satisfied by the information contributed in the side-issue appositive, that John has another vehicle. This seems to contradict the restriction that Potts imposes on the direction of the flow of information, if we assume a loose, but reasonable, interpretation of what 'flow of information' is and take it to include the satisfaction of a presupposition.

In this paper we argue for a treatment of conventional implicature in terms of multiple dimensions, in agreement with Potts (2005, 2007) and *contra* AnderBois et al. (2010). To explain the data discussed by AnderBois et al. (2010) we will propose a two stages analysis: during the compositional process the two dimensions are kept separate and obey Potts's restrictions regarding the flow of information, while in the post-compositional phase (i.e. when anaphoric relations are established and discourse consistency checks are performed) the interactions between the dimensions are free.

The main challenge is to create a compositional model of this procedure and the main contribution of this paper is the introduction of such a model. We propose to use a mathematical construction known as *monads* already used in the formal semantics of programming languages, and introduced to linguistics by Shan (2001), to model an impressively wide array of natural language semantics phenomena. Here we show how monads can be used to analyse conventional implicature as a purely compositional process. At the same time we show how monads can be used to create meaning terms that include the information necessary to predict the correct restrictions on the results of the second stage of our analysis, the one involving anaphora resolution and discourse consistency checks. We will see that monads offer all the necessary machinery:

- they allow us to group together multiple semantic objects while retaining the same compositional structure (in the sense of the same proof theoretical object),
- they allow us to enforce a specific order of evaluation, necessary to identify acceptable uses of co-reference (meant here in a broad sense).

However we will also argue that the main advantage offered by monads is that of a principled and generalized explanation. As already mentioned, Shan (2001) showed how monads can be used to model a number of semantic phenomena ranging from focus, to question semantics and scope, and more recently Giorgolo and Unger (2009) used them to model anaphora. We interpret this flexibility as an indication that monads capture some deep structure of natural language semantics, which occurs over and over again. We will argue that the mathematical components of the definition of a monad (the *functor* and the two *natural transformations*) are the heart of this notion of occurrence.

We will couch our monad analysis within Glue Semantics (Dalrymple et al., 1993, Dalrymple, 1999, 2001, Asudeh, 2012). This approach to treating conventional implicature in a compositional fashion has antecedents. Potts (2005) proposes an alternative version of his theory in terms of Glue Semantics. This implementation has been refined and fully incorporated in the Lexical-Functional Grammar (LFG) framework by Arnold and Sadler (2010, 2011). These latter proposals are strongly tied to the LFG framework and depend on a number of LFG-specific assumptions. A more general proposal is the one of Barker et al. (2010). Their model has many points of contact with the one presented here and shares very similar goals. Barker et al. use a continuation-based approach to model the restricted interaction between dimensions. While continuations are provably equivalent to the general monadic framework we use for our model, we will argue that our approach is preferable in that it zeroes in on the essential properties of conventional implicature and predicts the full range of interactions without having to depend on the full power of monads.

The paper is structured as follows. In section 2 we review the arguments in favor and against a multidimensional semantics for conventional implicature and show why a multidimensional semantics is indeed necessary. In section 3 we review two previous proposals to model the interaction between dimensions in a compositional fashion. Section 4 introduces the technical machinery behind our analysis, which is exemplified in section 5 with a fully worked out semantic derivation. We conclude in section 6 with some closing remarks.

2 Interdimensional Meaning Interaction

We have already discussed that AnderBois et al. (2010) reject the Potts's multidimensional analysis on the basis of a number of circumstances, initially discussed by Potts (2005:52ff.), in which at-issue content seems to require access to side-issue content. We repeat here some of the cases they analyze:

1. **Presupposition**

   (4) Mary, a good drummer, is a good singer <u>too</u>.

   This example is comparable to the one previously discussed. Here the presupposition that Mary has some additional musical talent besides being a good singer is supported by the information conveyed by the nominal appositive *a good drummer*
2. **Anaphora**

   (5) Jake$_1$, who almost killed a woman$_2$ with <u>his</u>$_1$ car, visited <u>her</u>$_2$ in the hospital.

   In (5) the pronoun *her* founds its antecedent in the non-restrictive relative clause, in a way leaking information from the CI dimension to the at-issue dimension.
3. **VP ellipsis**

   (6) Lucy, who doesn't help her sister, told Jane <u>to</u>.

   Similarly here the elided VP is first introduced in the relative clause, i.e. in the CI dimension.
4. **Nominal ellipsis/anaphora**

   (7) Melinda, who won three games of tennis, lost because Betty won <u>six</u>.

   Also here the nominal ellipsis (or the pronominal use of *six*) seems to break the Pottsian rule of information flowing only from the at-issue to the CI dimension.

According to AnderBois et al. (2010) this kind of data makes the analysis of Potts untenable. The reasoning is that given that the multidimensional treatment of conventional implicatures is founded on the intuition that the dimensions are fundamentally independent (or at least the at-issue dimension is independent of the side-issue one) there is no way to reconcile it with the type of data just discussed. Their conclusion is therefore that there is only *one dimension* of meaning and that conventional implicatures live happily together with at-issue meaning in this dimension.

Clearly a purely one-dimensional approach is faced with major challenges when trying to explain the other kind of data, exemplified in the introduction by the two mini-dialogues (1) and (2). In a unidimensional approach there is in fact nothing preventing side-issue content from interacting with logical operators such as negation. The solution that AnderBois et al. propose is that, instead of two dimension of meaning, there are *two modes* of discourse update, one for at-issue material and one for side-issue material. At-issue material is *proposed* and open for correction, questioning, etc. Side-issue material is instead *imposed* and the update eliminates possible interpretations that are inconsistent with the side-issue meaning. In this way, the data in (4–7) is explained on the basis of a full interaction between dimensions, and at the same time the fact that in (1) and (2) the replies of B do not target the side-issue comments is accounted for on the ground of its imposed status.

Our understanding is that this analysis is however not capable of accounting for a number of other circumstances. The first problem we encounter when trying to apply AnderBois et al. is when conventional implicatures interact with logical operators that are not at the discourse level. AnderBois et al. (2010) do not explain how discourse updates are generated from syntactic structures but given their assumption of unidimensionality we assume that a single logical form is derived. In light of this hypothesis consider the following example:

(8) Luke Skywalker is so gullible that he believes that Jabba the Hutt, a notorious scammer, is a trustworthy business partner.

Here clearly the information that Jabba is a scammer is not part of Skywalker's beliefs, therefore projecting somehow outside of the scope of *believes*. To obtain such an effect with a single logical form we would have to postulate an *ad hoc* rule that moves the content of the appositive outside of any logical operator above it in the derivation. Similar rules would be needed for all other expressions conveying conventional implicatures leading to an undesirable proliferation of scope-escaping devices. Alternatively the appositive could be somehow marked in the logical form as requiring special treatment during discourse update, but then it would be difficult to see how such an approach would be relevantly different from Potts's analysis.

A unidimensional approach also seems incapable of correctly predicting the patterns of interaction between the at-issue and side-issue meaning. The interactions are in fact not completely free, but instead seem to respect limitations that we observe at the discourse level. This is also true for the cases in which information flows from the at-issue to the side-issue dimensions/modes. Consider the following examples:

(9) All Cairo taxi drivers$_1$, who by the way painted their$_1$ taxis red in protest, are on strike.

(10) *Every Cairo taxi driver$_1$, who by the way would threaten me with his$_1$ gun, is on strike.

If we assume a single logical form such that both relative clauses are in the scope of the respective quantifiers, then it is not quite clear why the possessive pronoun *their* in (9) can be bound by the quantifier *all*, while the same is not true in (10). What we observe here seems to be more in line with an analysis in which the two pronouns are considered anaphoric (i.e. as part of two distinct sentences), so that their acceptability is dependent on their ability to establish an anaphoric relation with their antecedents. In the case of (9) *their* manages to find its antecedent (the totality of Cairo taxi drivers) thanks to the well known ability of plurals to establish discourse referents (Nouwen, 2003). In (10) instead the pronoun *his* cannot be anaphorically related to referent introduced by the universal quantifier as it is local to its scope.

We propose a different solution based on the observation that the interaction between at-issue and side-issue content is limited to a certain class of discourse related phenomena and follows the same patterns we observe when dealing with proper discourse fragments. We start from the intuition that at the level of logical form generation we have two distinct semantic dimensions. The interaction between these two dimensions follow Potts's principle of limited interaction: at-issue meaning resources can be re-used in the CI dimension but side-issue content never leaks into the at-issue dimension. Once the logical form is complete (and possibly contains things like free variables and presuppositions to be satisfied) the boundaries between dimensions are lifted and the resolution of discourse-related uncertainties can take place, under the condition that we have two propositions that function at all effects as two distinct discourse segments.

This analysis correctly accounts for the data. The non-interaction between conventional implicatures and logical operators is explained, following Potts, in terms of multidimensionality. For instance if we apply our analysis to example (8) we obtain two distinct propositions, one expressing the fact that Skywalker is so gullible that he believes that Jabba is a good business partner and the other expressing the fact that Jabba is a notorious scammer. The limited interactions we observe in examples (4–7) are instead explained as a discourse-level phenomena. This also allows us to predict the acceptability of (9) and the non-acceptability of (10).

The next step is to explain how we can build in a compositional fashion the two meaning components while keeping the flow of information under control. Before describing our approach we will review two previous proposals in further detail.

3 Conventional Implicature and Compositionality

The analysis by Barker et al. (2010) builds on the idea of continuation-based semantics, a promising approach explored by the authors in a number of settings. The use of continuations allows them to model with a single device the full stack of operations needed to account for the data under discussion. The analysis is quite complex and we report here only the main characteristics of it.

The general idea is that the interpretation of a sentence is a function not only of its component parts but also of the speaker uttering it (for instance necessary to obtain the correct attribution of judgements in case of expressives) and the current common ground, i.e. the collection of propositions introduced in the discourse up to the expression under consideration. The output of the interpretation of a sentence is a pair of propositions that correspond to the at-issue and side-issue contributions to the common ground. Therefore the *return type* of their semantic objects is a function from a speaker to a function from an input common ground to a pair of propositions.

The lexical meanings are lifted to a continuation-passing form, where each semantic object is wrapped into a function that takes its context as an argument and applies the original meaning to it as its argument. The meaning of at-issue expressions is however in a way different from that of expressions conveying conventional implicatures. In fact, in the first case the return type of the context is opaque to the lexical entry; that is, the lexical entry does not contain the information that it is a function from a speaker to a function from a common ground to a pair of propositions. The return type is instead part of the lexical item that contributes to the CI dimension. In this way, these lexical item can operate on it and, for example, contribute to the proposition corresponding to the CI dimension. Therefore in Barker et al. (2010)'s approach the flow of information is completely controlled at the level of types.

However, the system loses part of its impressive elegance due to two minor drawbacks:

1. Barker et al., following Potts (2005), try to restrict the space of possible lexical items operating on the side-issue dimension. Their main concern is to prevent the existence of lexical items that modify the side-issue component accumulated so far, as in the following example:

   (11) John <u>negex</u> read the damn book.
   Side-issue: John feels good about the book.

In order to prevent the existence of items like **negex**, they require lexical items to satisfy a theorem stating the order independence of the computation of side-issues with respect to the update function. In order to correctly prevent the problematic modifiers, they need to stipulate a restriction on the possible context update functions. Without this stipulation, the continuation-based semantics does not provide enough structure to exclude the existence of the impossible modifiers.

2. The system, as it stands, does not support simultaneous treatment of quantification and multidimensional meaning. The treatment of quantification requires the result type of the continuation to be the type of truth values, whereas the treatment of multidimensionality requires the result type to be a function from a speaker to an input context to a pair of types for truth values.

This second drawback is related to the fact that the operations necessary to account for the various semantic phenomena in the continuation-passing approach are all dependent on the single context passed as an extra argument to each meaning. The monadic approach does not encounter this problem. In fact, although the two approaches are equivalent (Wadler, 1992), each single monad is capable of simulating only a single linguistic side-effect. Therefore, by showing that a semantic phenomenon can be modelled as a monad, we manage, at the same time, to be more precise about the specific operations that are required and to isolate these operations from those necessary to model concurrent phenomena. The price we pay for this clear distinction between phenomena is a more complex way of combining them. We will see below that to actually combine monads we have to leave the nice mathematical framework they offer and move to a different construct known in computer science as a *monad transformer*.[1]

In our opinion, the advantage of a monadic approach over a continuation-based one is not only a matter of mental hygiene. We believe the monadic approach better captures the idea that what we observe in natural language is the reuse of the same fundamental patterns over and over again, applied to different settings but based on the same underlying principles.

Another approach to modeling the composition of at-issue and side-issue meaning is the one of Arnold and Sadler (2010, 2011). Arnold and Sadler's analysis is cast in the framework of LFG and follows a suggestion by Potts (2005:85ff.) in capturing multidimensionality directly in the logic for composition. In the context of the glue logic of LFG's Glue Semantics, where the glue logic is a non-commutative fragment of linear logic, this means that, not dissimilarly to what Barker et al. (2010) do, the glue logic terms on the right side of glue meaning constructors are used to distinguish lexical resources that contribute different types of semantic content. For readers not familiar with Glue Semantics, this means that the job controlling the flow of information is deferred to the types and the logic determining their composition. In contrast we will see that the monadic approach leaves the type signature of our lexical items (superficially) unchanged and moves the responsibility of keeping track of the information flow to the meaning terms. The proposal by Arnold and Sadler also presents a number of downsides mainly related to the limited expressive power of the logical terms of Glue Semantics. In particular the problems are all related to the fact that to implement the coupling of at-issue and CI material we have to resort to a *tensor product* connective that presents a number of problems:

[1]A more promising approach that we are exploring is to use simpler objects, known as *applicative functors*, that can be composed freely.

1. In principle, it might be necessary to propose more than two dimensions. In such a case, the commutative tensor conjunction in linear logic does not provide enough structure to properly distinguish between dimensions or to refer to information in a particular dimension subsequently.
2. The lack of structure in the tensor conjunction makes it difficult to control at-issue/side-issue interactions of the kind discussed above.
3. Tensors in proof goals make it more difficult to state the correct condition on proof termination and therefore potentially lose some of the linguistic leverage provided by linear logic's resource sensitivity (Asudeh, 2004, 2012).

4 Monads for Conventional Implicature

In this section we introduce the technical machinery implementing our analysis. The main ingredients of our implementation are the notion of *paired semantic values* as introduced by Potts (2005) and the monadic interface proposed by Shan (2001).

According to Potts, expressions conveying conventional implicatures denote two semantic objects: an at-issue value (which is often empty and corresponds to the identity function) and a side-issue component which is always a proposition. This assumption will also form the core of our approach. The difference is that in our approach all expressions are interpreted as denoting a pair of values. The first component of the pair denotes the at-issue contribution of the expression, while the second component is not a proposition but rather a *collection of propositions*, containing all the side-issue information conveyed by the sub-parts of which the expression is composed. Expressions without conventional implicature-bearing items denote an empty collection of CI propositions. The monadic framework allows us to reuse the standard compositional machinery to compose these more complex meanings, while controlling the flow of information as desired.

Monads originated in *category theory*, a very general mathematical theory about structures and mappings between them, intended in the broadest sense. They have found a successful application in computer science in the field of programming language semantics and as a way to structure functional programs. To get a good understanding of how we propose to use monads to deal with conventional implicatures, we will use intuition stemming from the use of monads both in category theory and computer science. We expect no previous knowledge of category theory, and, although we will use some categorical notions, we will keep the discussion at an intuitive level.

Monads come in different forms, and different mathematical constructions can be understood as following the monadic pattern. However, they can all be defined in terms of three objects: a *functor* and two *natural transformations*. The idea behind a functor is that of a mapping between two structures (intended as a collection of objects and structuring connections between them) that reflects the pattern of connections of the source structure in the target structure. A natural transformation is a more complex concept, but intuitively it can be understood as a way to transform objects that are both images of the same starting object under two different functors (having the same target structure) in such a way that the properties of the first structure are conserved in both mappings to the target structure. In the case of a monad the first functor is the identity one, intuitively a functor that does nothing, so for our purposes a natural transformation can be understood as an additional condition of similarity between the original structure and the target one identified by the functor.

Another way to understand what a monad does is to think of it as a way to reproduce the structure of a space of values and functions in a richer setting that carries more information, in the sense that we can specify more things about the values and functions. The idea is that we can move from the information-poor space to the information-rich space by mapping a value or function in the poor space to an information-enriched counterpart (in a sense this is what the monad's functor does). We do so by associating the value or function with some sort of default information. In this way, we get an object of the right information-rich type, without committing to any particular enriched information (respecting the conditions imposed by the natural transformations).

More concretely, we use monads to create a structured mapping between the space of standard isolated semantic values (i.e. things of type e, t, $e \to t$, etc.) to the richer space of paired values, where the first component is any value and the second one is some type of collection (represented as a monoid) of propositions.[2] In this case the role of the default information is played by the empty collection corresponding to the case when no conventional implicature is expressed.

Another useful way of looking at monads is in terms of their ability to structure computations. A monad is considered a generic computation that yields a value and that can perform some other operations in the background, the *side effects* of a computation. Shan's (2001) intuition is that we can model many (apparently) non-compositional phenomena as side effects of computing the main value of an expression. Similarly we can consider conventional implicatures as side effects. Specifically, expressions contributing to the CI dimension can be seen as computations that, besides yielding a (possibly empty) value usable in the at-issue dimension, *log* some additional information to a special place, the CI dimension. The operation of logging information is well known in computer science and it is normally modelled in terms of a monad known as the *Writer* monad. We will use in the rest of the presentation this metaphor of a writing device to make clear how we intend to use monads to model conventional implicature.

The monad we will use can be defined in terms of a triple $\langle M, \eta, \star \rangle$. M, the functor, brings us from the unary values to the paired ones. It is at the same time a label for these rich values and it can in principle be unwrapped as a product type made of an arbitrary type and the type of collections of propositions. η ('unit') is the natural transformation that tells us how the single

[2]For readers with some background in category theory, the endofunctor of our monad goes from the cartesian closed category of standard Montagovian types enriched with the propositional monoids to the subcategory whose objects are products of any value and a monoid, and the arrows are only the arrows that qualify as *isotone* with respect to the second component of the pair, where an arrow f is isotone in this sense if it satisfies the following condition:

$$f(x) = y \text{ such that } \pi_2(x) \leq \pi_2(y) \tag{12}$$

π_2 is the projection extracting the second component of the pair, while $\leq$ is one of the pre-orders that we can associate with a monoid.

Every monoid gives rise to two pre-orders $\leq_{\text{pre}}$ and $\leq_{\text{post}}$ defined as follows:

1. $x \leq_{\text{pre}} y$ iff $\exists z. z \cdot x = y$
2. $x \leq_{\text{post}} y$ iff $\exists z. x \cdot z = y$

Proposition 1. $\leq_{\text{pre}}$ *and* $\leq_{\text{post}}$ *are pre-orders*

Proof. The existence of the identity element e guarantees reflexivity, as for all x, y $e \cdot x = x$, $x \cdot e = x$. For transitivity we consider two cases: (1) assume that $x \leq_{\text{pre}} y$ and $y \leq_{\text{pre}} z$, i.e. $k \cdot x = y$ and $h \cdot y = z$, then we have $h \cdot (k \cdot x) = z$ and by associativity of $\cdot$ we have also that $(h \cdot k) \cdot x = z$, i.e. $x \leq_{\text{pre}} z$; (2) assume that $x \leq_{\text{post}} y$ and $y \leq_{\text{post}} z$, i.e. $x \cdot k = y$ and $y \cdot h = z$, then we have $(x \cdot k) \cdot h = z$ and by associativity of $\cdot$ we have also that $x \cdot (k \cdot h) = z$, i.e. $x \leq_{\text{post}} z$. $\square$

values are to be mapped in a consistent way into paired values. This is the operation we will use to lift the standard meaning into the richer setting of paired values. To do so we simply couple the original meaning with the empty collection of propositions. To simplify things, we will fix the type of the second component of the pair to be the type of sets of propositions. $\star$ ('bind') has a more computational interpretation. We can think of it as the mechanism for extracting values from computations and creating new computations using these values. $\star$ also allows ordering for side-effects of computations. $\star$ offers a way to compose computations in an ordered fashion. In the case of our *Writer* monad, $\star$ is implemented as a binary function that takes 1) an input pair of a variable and a collection of propositions and 2) a function f that produces a computation. $\star$ produces a new computation whose value is the value of the computation produced by f and a new collection of propositions that is the union of the input collection of propositions with the collection of propositions produced by f. Formally:

$$\langle x, P\rangle \star f = \langle \pi_1(f\,x), P \cup \pi_2(f\,x)\rangle \tag{13}$$

It is this function that has the role of threading the collection of propositions through the derivation. Lexical items have no control over the threading of information in the CI dimension; they cannot block it or operate on it beside writing new information. Once the compositional process ends, the information collected in the at-issue and the CI dimension are fully exposed and become available for further processing.

To effectively compose monadic meanings we have to show we can associate them with terms of semantic derivation. We will work in the setting of Glue Logic, but the definitions we present here can be trivially adapted to any other type-logical framework. In the Glue setting, we want to keep as much as we can of the standard glue logic, but use the mapping facility of monads to obtain the additional side-issue dimension. This means that we will restrict ourselves to an implicational fragment of linear logic. However we need to distinguish between two different modes of composition. On the one hand, we have standard at-issue-only lexical items that are oblivious to the enriched setting in which they operate. From the perspective of these items only the first component of our enriched meanings count. On the other hand, we have expressions like appositives that instead make full use of the two dimensions by writing information in the second one, reusing values extracted from the at-issue component of the surrounding linguistic items. To reflect the two modes of composition, we introduce an additional implication in the logic corresponding to the more complex form of composition associated with conventional implicatures.

We present the logic of composition in terms of natural deduction elimination and introduction rules. For the standard linear implication, $\multimap$, the rule for elimination are the usual ones and are shown in (14).

$$\frac{x : A \qquad f : A \multimap B}{\mathcal{A}(f)(x) : B} \multimap E \qquad\qquad \frac{\begin{array}{c}[\eta(x) : A]_{\mathrm{i}} \\ \vdots \\ t : B\end{array}}{\eta(x) \lhd t : A \multimap B} \multimap I_{\mathrm{i}} \tag{14}$$

The proof terms corresponding to the two rules require some explanation. In the case of the elimination rule we use the following special form of function application introduced by Shan (2001):

$$\mathcal{A}(f)(x) =_{\mathrm{def}} f \star \lambda g.x \star \lambda y.\eta\,(g\,y) : M\,(\alpha \to \beta) \to M\,\alpha \to M\,\beta \tag{15}$$

This operation takes as its arguments a monad that yields a function of type $\alpha \to \beta$ as its return value and a monad that returns a type α. It then runs in sequence the first computation binding the result to the variable g and the second one binding its return value to y, returning the result of applying g to y (in the usual sense) wrapped in a new computation that adds nothing new. In the background, $\star$ takes care of threading the (possible) conventional implicatures associated with f and x or their sub-expressions.

In the case of the introduction rule, a hypothetical resource corresponds to an innocuous computation that yields a hypothetical value, which is later retracted. The meaning of the term $\eta(x) \lhd t$ is given by the following equation:

$$\eta(x) \lhd m =_{\text{def}} m \star \lambda b.\eta\,(\lambda x.b) : M\,\alpha \to M\,\beta \to M\,(\alpha \to \beta) \tag{16}$$

where x must be a fresh variable not appearing anywhere else in the proof.[3]

The second type of implication $\multimap_*$ has very similar elimination and introduction rules:

$$\frac{x : A \qquad f : A \multimap_* B}{A_*(f)(x) : B} \multimap_* E \qquad \frac{\begin{array}{c}[x:A]_{\mathrm{i}}\\ \vdots\\ t : B\end{array}}{x \lhd_* t : A \multimap_* B} \multimap_* I_{\mathrm{i}} \tag{17}$$

In this case the proof terms corresponding to the application of these two rules have a much simpler interpretation. A_* corresponds to standard function application, but it is restricted to paired objects:

$$A_*(f)(x) =_{\text{def}} f\,x : (M\,\alpha \to M\,\beta) \to M\,\alpha \to M\,\beta \tag{18}$$

$x \lhd_* m$ is equivalent to standard abstraction:

$$x \lhd_* m =_{\text{def}} \lambda x.m : M\,\alpha \to M\,\beta \to (M\,\alpha \to M\,\beta) \tag{19}$$

In the next section, we put this machinery into action.

5 Analysis

We work through the following very simple example:

(20) John, who likes cats, likes dogs also.

Here we have a non-restrictive relative clause whose content is necessary to satisfy the presupposition introduced by *also*. We can construct a semantic representation that keeps the at-issue and CI-dimensions separated and at the same time accumulates the condition imposed by the presupposition trigger in a third independent location. To do so we combine two *Writer* monads, which requires a monad transformer. We skip over the details of these here; the reader can find a short introduction in the appendix and a fuller discussion of transformers in Shan (2001).

Lexicon

We assume fairly standard lexical entries. All the entries for lexical items not introducing conventional implicatures or presupposition are simply lifted version (through the use of η) of

[3]Here we are perhaps overly cautious, but in this way we never risk binding a free variable.

comma	$\lambda j \lambda l. j \star \lambda x. l \star \lambda f.\texttt{write}(f\ x) \star \lambda\_ . \eta(x) : j \multimap_* (j \multimap l) \multimap_* j$
also	$\lambda v. \lambda o. \lambda s. s \star \lambda x. v \star \lambda f. o \star \lambda y.\texttt{check}(\exists z. f\ z\ x \wedge z \neq y) \star \lambda\_ . \eta(f\ y\ x) :$ $(d \multimap j \multimap l) \multimap_* d \multimap_* j \multimap_* l$
John	$\eta(j) : j$
who	$\eta(\lambda P.P) : (j \multimap l) \multimap (j \multimap l)$
likes	$\eta(\lambda y \lambda x. like(x,y)) : c \multimap j \multimap l$
cats	$\eta(\iota x. cat^*(x)) : c$
likes	$\eta(\lambda y \lambda x. like(x,y)) : d \multimap j \multimap l$
dogs	$\eta(\iota x. dog^*(x)) : d$

Table 1: Lexicon for *John, who likes cats, likes dogs also.*

what we would expect in a traditional lexicon. The prosodic element introducing the non restrictive relative clause (here represented as *comma*, following Potts 2005, 2007) and the presupposition trigger instead have more complex entries. The lexicon is specified in Table 1. The lexical entries of *comma* and *also* are dependent on the surface order of their respective arguments. Information about linear order can be fed to the semantic derivation as in Asudeh (2009).

Both `write` and `check` are monadic functions recording a proposition to two different logging storages. We use `write` to add propositions to the CI dimension. `check` is used to record the presuppositional condition that must be checked in the post-compositional phase.[4] The two functions have type $t \rightarrow M\ \perp$, where $\perp$ is a type with the single inhabitant $\perp$, and they are both defined as follows:

$$\lambda t. \langle \perp, \{t\} \rangle \tag{21}$$

Proof

The proof for example (20) is shown in Figure 1. We split the proof into two sub-proofs for presentational purposes; the sub-proofs are connected at the point marked $\boxed{1}$. The result is a pair whose first member is in turn another pair. The second component of the outer pair is the collection of conditions on the common ground required by the presuppositional items. The first component of the inner pair represents the at-issue meaning, namely that John likes dogs, while the second member represents the collection of side-issue contributions so far, namely that John likes cats. The presuppositional condition imposed by **also** is satisfied by the side-issue contribution $like(j, \iota x. cat^*(x))$. Most importantly, the information necessary to compute the satisfaction of the presupposition becomes accessible only at the end of the compositional process, since the log produced by `write` cannot be examined before the monadic computation terminates.

[4]To be precise, `check` must be *lifted* to the monad transformer corresponding to the side-issue logging system. Therefore `check` should be read as `lift(check)` where `lift` is the function that lifts a monadic computation to a monadic transformer level (see the appendix). For the case discussed here `lift` can be implemented as follows:

$$\texttt{lift}(m) = m \star \lambda x. \eta(\langle x, \{\ \} \rangle)$$

$$\dfrac{\dfrac{\dfrac{[\![\text{john}]\!]}{j} \quad \dfrac{[\![\text{comma}]\!]}{j \multimap_* (j \multimap l) \multimap_* j}}{(j \multimap l) \multimap_* j}\multimap_* E \qquad \dfrac{\dfrac{[\![\text{who}]\!]}{(j \multimap l) \multimap (j \multimap l)} \quad \dfrac{\dfrac{[\![\text{likes}]\!]}{c \multimap j \multimap l} \quad \dfrac{[\![\text{cats}]\!]}{c}}{j \multimap l}\multimap E}{j \multimap l}\multimap E}{\boxed{1}\ j}\multimap_* E$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[\![\text{also}]\!]}{(d \multimap j \multimap l) \multimap_* d \multimap_* j \multimap_* l} \quad \dfrac{[\![\text{likes}]\!]}{d \multimap j \multimap l}}{d \multimap_* j \multimap_* l}\multimap_* E \quad \dfrac{[\![\text{dogs}]\!]}{d}}{j \multimap_* l}\multimap_* E \quad \boxed{1}}{l}\multimap_* E}{\langle\langle like(j, \iota x.dog^*(x)), \{like(j, \iota x.cat^*(x))\}\rangle, \{\exists z.like(j,z) \wedge z \neq \iota x.dog^*(x)\}\rangle : l}$$

Figure 1: Proof for *John, who likes cats, likes dogs also.*

6 Conclusion

We have presented a fully compositional analysis of conventional implicatures. We started by discussing the necessity of keeping the at-issue and side-issue components of meaning separated during the compositional process. Our analysis is based on the idea that the correct model of conventional implicature requires two stages: a compositional one, during which interactions are limited, and a post-compositional one, in which the interactions are freer and are governed by the same principles regulating the relations between distinct discourse segments; this latter level models the freer interactions discussed by AnderBois et al. (2010). In contrast, having the two components in the same dimension does in fact not solve the problem of limited interaction we observe when composing meaning.

We presented a fully implemented model of our analysis. The analysis makes use of monads as a way to structure, in a uniform compositional way, different putatively non-compositional semantic phenomena. Our approach offers a number of advantages over similar ones, especially in terms of economy of theoretical assumptions and in generality of the type system.

More interestingly, we think that the monadic approach, and similar categorical constructions, allow us to have a clearer understanding of the theoretical objects we deal with in formal semantics, and begins to give us a way to see how these same structures and procedures are put to use cognitively in different contexts, given the normal understanding of cognition as computation.

References

AnderBois, Scott, Adrian Brasoveanu, and Robert Henderson. 2010. Crossing the appositive/at-issue meaning boundary. In *Proceedings of SALT 20*, ed. Man Li and David Lutz, 328–346.

Arnold, Doug, and Louisa Sadler. 2010. Pottsian LFG. In *Proceedings of LFG10*, ed. Miriam Butt and Tracy Holloway King, 43–63. Stanford, CA: CSLI Publications.

Arnold, Doug, and Louisa Sadler. 2011. Resource splitting and reintegration with supplementals. In *Proceedings of LFG11*, ed. Miriam Butt and Tracy Holloway King, 26–46. Stanford, CA: CSLI Publications.

Asudeh, Ash. 2004. Resumption as resource management. Doctoral Dissertation, Stanford University.

Asudeh, Ash. 2009. Adjacency and locality: A constraint-based analysis of complementizer-adjacent extraction. In *Proceedings of the LFG09 conference*, ed. Miriam Butt and Tracy Holloway King. Stanford, CA: CSLI Publications.

Asudeh, Ash. 2012. *The logic of pronominal resumption*. Oxford: Oxford University Press.

Barker, Chris, Raffaella Bernardi, and Chung chieh Shan. 2010. Principles of interdimensional meaning interaction. In *Proceedings of SALT 20*, ed. Man Li and David Lutz, 109–127.

Dalrymple, Mary, ed. 1999. *Semantics and syntax in Lexical Functional Grammar: The resource logic approach*. Cambridge, MA: MIT Press.

Dalrymple, Mary. 2001. *Lexical Functional Grammar*. San Diego, CA: Academic Press.

Dalrymple, Mary, John Lamping, and Vijay Saraswat. 1993. LFG semantics via constraints. In *Proceedings of the Sixth Meeting of the European ACL*, 97–105. University of Utrecht: European Chapter of the Association for Computational Linguistics.

Giorgolo, Gianluca, and Christina Unger. 2009. Coreference without discourse referents: A non-representational drt-like discourse semantics. In *Computational Linguistics in the Netherlands*, ed. B. Plank, T. Kim Sang, and T. Van de Cruys, Vol. 4 of *LOT Occasional Series*, 69–81.

Nouwen, Rick. 2003. Plural pronominal anaphora in context. Doctoral Dissertation, Utrecht Institute for Linguistics OTS.

Potts, Christopher. 2005. *The logic of conventional implicatures*. Oxford: Oxford University Press.

Potts, Christopher. 2007. The expressive dimension. *Theoretical Linguistics* 33:165–197.

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

Wadler, Philip. 1992. The essence of functional programming. In *Proceedings of the 19th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, POPL '92, 1–14. New York, NY, USA: ACM.

A Monad Transformers

In general it is not possible to combine two monads $(M_1, \eta_1, \star_1)$ and $(M_2, \eta_2, \star_2)$ to get a third monad $(M_1 \circ M_2, \eta_1 \circ \eta_2, \star_1 \circ \star_2)$.

The solution is to "lift" the monadic mappings to operate directly on informationally rich meaning spaces.

From each monad we (mechanically) generate a *monad transformer*. The monad transformer encapsulates the same type of computation performed by the original monad (writing/reading from a global state, generating a value in a non deterministic way, etc.). However, rather than mapping from the value space (the informationally poor meaning space) to the monadic space, we create a mapping from another monadic (rich) space to the one representing the computation we are interested in. Effectively, each monad transformer can be seen as a collection of monads distinguished by the monadic space from which they map.

Monad transformers are monads; thus their definition is given in terms of the standard operations η and $\star$. However, we also need an additional operation, usually called `lift` and with type $M\,x \to MT\,M\,x$, where M is the monad indexing the specific instance of the monad transformer MT. The function `lift` maps a specific instance of a monadic rich value to an even richer one in the space defined by the monad transformer.