$\langle M, \eta, \star \rangle$

Gianluca Giorgolo
Gianluca_Giorgolo@carleton.ca
Carleton University

Ash Asudeh
ash.asudeh@ling-phil.ox.ac.uk
Carleton University &
University of Oxford

September 6, 2011
Sinn und Bedeutung 16 · University of Utrecht

1 Introduction

(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

- Potts (2005, 2007) has claimed that the analysis of these sentences requires two semantic levels, typically called 'dimensions'.

  1. The 'at-issue' dimension represents the aspect of meaning that is under discussion and is sensitive to logical operators such as negation.
  2. The 'side-issue' dimension (né 'CI dimension') represents an aspect of meaning that contributes information that is speaker-oriented, often peripheral, and not under discussion or up for grabs.

- In example (1), the fact that the speaker hates dogs and/or their defecatory habits cannot be controversial: the speaker communicates this by his/her choice of words.

- A crucial aspect of Potts's multidimensional type theory is that information can flow from at-issue content to side-issue content, but not vice versa.

This research is supported by an Early Researcher Award from the Ontario Ministry of Research and Innovation and NSERC Discovery Grant #371969.

- AnderBois et al. (2010) have claimed that multidimensional semantics for at-issue/side-issue meanings is problematic, because of example like the following.

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

  – Here we seem to see full interaction between the two dimensions. An anaphor in the side-issue appositive relative is finding its antecedent in the main at-issue content. This is to be expected on the standard account. 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.

- Barker et al. (2010) share our goal of giving a compositional treatment for conventional implicatures. Their solution employs a continuation-based semantics for categorial grammar. However, they run into two serious limitations — one having to do with the inability to block a certain type of interaction between at-issue and side-issue content and the other having to do with the simultaneous treatment of multidimensionality and quantifier scope, as exemplified by example (1) above.

2 Main Claims

- A multidimensional treatment of conventional implicature (appositives, expressives, etc.) is necessary (in agreement with Potts).
- Our intuition is that at-issue and side-issue meaning are generally separate, but there are limited interactions (contra AnderBois et al.).
- Monads are a good language for expressing multidimensionality in the meaning language while capturing limited interaction between dimensions. The following properties of monads are of particular interest in this respect:
  1. Monads can be used as containers for more than one value.
  2. Monads can be used to impose an order of evaluation.
- The monadic treatment does not incur the two problems mentioned above that are encountered by Barker et al. (2010).

3 Overview

1. *Introduction*
2. *Main Claims*
3. *Overview*
4. Interdimensional Meaning Interaction
5. Background: Monads
6. Analysis
7. Conclusion

4 Interdimensional Meaning Interaction

- AnderBois et al. (2010) review a number of circumstances, initially discussed by Potts (2005: 52ff.), in which at-issue content seems to require access to side-issue content, which would be precluded by Potts's type theory.

  1. Presupposition

     (4) Mary, a good drummer, is a good singer too.

  2. Anaphora

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

  3. VP ellipsis

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

  4. Nominal ellipsis

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

- AnderBois et al. (2010) conclude from this kind of data that there is only *one dimension* of meaning, since there seems to be interaction between the at-issue and side-issue meanings and the multidimensional treatment is founded on an intuition of independence.

- AnderBois et al. instead propose that 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.

- Barker et al. (2010) start from the same dual-update perspective as AnderBois et al. (2010), but add a specification of how the updates are compositionally calculated.

- The main properties of the Barker et al. system are:

  1. Continuations are used to represent the fact that side-issue meaning is dependent on the speaker of the utterance and an input context, represented as the characteristic function of a set of worlds.
  2. The continuation returns a pair of propositions (characteristic functions of sets of worlds).
  3. Interaction between at-issue and side-issue meaning is limited at the level of types by making at-issue interpretations polymorphic in the result type of the continuation, while side-issue interpretations exploit the specific form of the result type of the continuation.

- While Barker et al.'s continuation-based system is similar to our monad-based system, their system faces two problems that ours does not:

  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 the following example:

     (8) John negex 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.

- The job of dimensions in Potts's theory, and also in our version, is to keep track of how information was introduced (as at-issue or side-issue contributions). AnderBois et al.'s proposal hardwires this distinction into discourse update, and therefore effectively fails to eliminate the bipartite nature of the Potts theory. This is reflected in the Barker et al. (2010) system, as they explicitly use a paired return type to distinguish between different forms of context update.

- Our intuition is instead that at-issue and side-issue content are largely separate, but that at-issue content can access side-issue content in certain limited circumstances. The interaction seems to be limited to post-compositional phenomena (e.g. satisfaction of a presupposition, resolution of anaphora and ellipsis)

- This intuition is motivated by the fact that side-issue content always ends up outside the scope of logical operators, such as negation, question-forming operators, etc., which was an important part of the initial motivation for Potts's claim of multidimensionality. On the AnderBois et al. theory, this lack of interaction with logical operators is unexplained.

- Assuming, then, that we wish to keep a multidimensional treatment, the next question is how to capture multidimensionality in a type-logical setting such as Glue Semantics.

- Arnold and Sadler (2010) follow Potts (2005: 85ff.) in capturing multidimensionality in the logic for composition. In the context of Glue Semantics, this means in the glue logic terms on the right side of glue meaning constructors.

- We propose an alternative approach in which multidimensionality is captured in the meaning language, while leaving the glue logic unidimensional, for the following reasons:

  1. In principle, it might be necessary to propose more than two dimension. 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).

- Monads provide a single mechanism for capturing multidimensionality and restricting interaction to a phase following composition.

5 Background: Monads

- Monads where first used to give a unified analysis of various semantic phenomena by Shan (2001)
- The main intuition behind monads is that they are 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.
- We can move from the information-poor space to the information-rich space as follows:
  - A value or function in the poor space is mapped to an information-enriched counterpart 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.
  - For example, in the case of multidimensionality, the values and functions that contribute only to at-issue material can be mapped to a richer space where they have a vacuous side-issue component.
- A more operational way to look at monads is to consider them as computations that yield values.
- A monad is defined by a triple $\langle M, \eta, \star \rangle$.
- η is the mapping from the information-poor space to the information-rich space.
- $\star$ is the mechanism for extracting values from computations and creating new computations using these values. $\star$ also allows ordering for side-effects of computation.
- M is the label for the information-rich counterpart of the original, information-poor types.
- For example, the *Writer* monad maps to an enriched type that pairs a value with a collection of propositions. For *Writer*:
  - η maps any value x to the pair $\langle x, \{\ \} \rangle$
  - $\star$ is 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:

    $$\langle x, P \rangle \star f = \langle \pi_1(f\ x), P \cup \pi_2(f\ x) \rangle \tag{9}$$
  - *Writer* therefore has the effect of logging a collection of propositions.
  - The values in *Writer* are restricted to computations that can only add values to the second component of the pair. The collection of propositions is not even accesible to monads layered on top of the *Writer* monad, effectively locking access to the collection during the compositional phase. In this way, items like **negex** are automatically precluded.
  - The result of the logging processes becomes available again once the compositional process reaches its end. We thus restrict interaction to a post-compositional phase.
- 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.
- The one augmentation to the glue logic that we require is an additional implication connective that allows some lexical items to be directly specified in terms of the information-rich space.

- This equation defines application for the standard glue logic implication, $\multimap$, in the monad-enriched meaning language.

$$A(f)(x) =_{def} f \star \lambda g.x \star \lambda y.\eta\ (g\ y) : M\ (\alpha \rightarrow \beta) \rightarrow M\ \alpha \rightarrow M\ \beta \tag{10}$$

- This equation defines abstraction for the standard glue logic implication, $\multimap$, in the monad-enriched meaning language.

$$\eta(x) \triangleleft m =_{def} m \star \lambda b.\eta\ (\lambda x.b) : M\ \alpha \rightarrow M\ \beta \rightarrow M\ (\alpha \rightarrow \beta) \tag{11}$$

  x must be a fresh variable not appearing anywhere else in the proof.

- These are the elimination and introduction rules, with a kind of Curry-Howard isomorphism to the monad-enriched meaning language, for the standard $\multimap$ implication.

$$\frac{x : A \qquad f : A \multimap B}{A(f)(x) : B} \multimap E \qquad \frac{\begin{array}{c}[\eta(x) : A]_i \\ \vdots \\ t : B\end{array}}{\eta(x) \triangleleft t : A \multimap B} \multimap I_i \tag{12}$$

- This equation defines application for the new glue logic implication, $\multimap_*$, in the monad-enriched meaning language.

$$A_*(f)(x) =_{def} f\ x : (M\ \alpha \rightarrow M\ \beta) \rightarrow M\ \alpha \rightarrow M\ \beta \tag{13}$$

- This equation defines abstraction for the new glue logic implication, $\multimap_*$, in the monad-enriched meaning language.

$$x \triangleleft_* m =_{def} \lambda x.m : M\ \alpha \rightarrow M\ \beta \rightarrow (M\ \alpha \rightarrow M\ \beta) \tag{14}$$

- These are the elimination and introduction rules, with a kind of Curry-Howard isomorphism to the monad-enriched meaning language, for the new $\multimap_*$ implication.

$$\frac{x : A \qquad f : A \multimap_* B}{A_*(f)(x) : B} \multimap_* E \qquad \frac{\begin{array}{c}[x : A]_i \\ \vdots \\ t : B\end{array}}{x \triangleleft_* t : A \multimap_* B} \multimap_* I_i \tag{15}$$

6 Analysis

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

Lexicon

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	$\lambda v.\lambda o.\lambda s.s \star \lambda x.v \star \lambda f.o \star \lambda y.\texttt{check}(\exists z.f\ x\ z \wedge z \neq y) \star \lambda\_.\eta(f\ x\ y) :$ $(d \multimap j \multimap l) \multimap_* d \multimap_* j \multimap_* l$
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	$\eta(\iota x.dog^*(x)) : d$

- The lexical entries of **comma** and **also** are dependent on the surface order of their respective arguments. It is possible to reshuffle the argument order without changing the semantic term's interpretation. The correct order can be selected by feeding information about linear order to the semantic derivation as discussed 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. [*]

- Assuming that W is the type-constructor/functor defining the *Writer* monad, the two functions have type $t \to W\ \bot$, where $\bot$ is a type with the single inhabitant $\bot$, and they are both defined as follows:

$$\lambda t.\langle \bot, \{t\} \rangle \tag{17}$$

Proof

The proof for example (16) is shown in Figure 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.

[*]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 Appendix). For the case discussed here `lift` can be implemented as follows:

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

$$
\dfrac{
  \dfrac{\llbracket \text{john} \rrbracket : j \qquad \llbracket \text{comma} \rrbracket : j \multimap_* (j \multimap_l) \multimap_* j}{A_*(\llbracket \text{comma} \rrbracket)(\llbracket \text{john} \rrbracket) : (j \multimap_l) \multimap_* j}
  \qquad
  \dfrac{\llbracket \text{who} \rrbracket : (j \multimap l) \multimap (j \multimap l) \qquad \dfrac{\llbracket \text{likes} \rrbracket : c \multimap j \multimap l \qquad \llbracket \text{cats} \rrbracket : c}{A(\llbracket \text{likes} \rrbracket)(\llbracket \text{cats} \rrbracket) : j \multimap l}}{A(\llbracket \text{who} \rrbracket)(A(\llbracket \text{likes} \rrbracket)(\llbracket \text{cats} \rrbracket)) : j \multimap l}
}{
  \boxed{1}\ A_*(A_*(\llbracket \text{comma} \rrbracket)(\llbracket \text{john} \rrbracket))(A(\llbracket \text{who} \rrbracket)(A(\llbracket \text{likes} \rrbracket)(\llbracket \text{cats} \rrbracket))) : j
}
$$

$$
\dfrac{
\dfrac{
  \dfrac{
    \dfrac{\llbracket \text{also} \rrbracket : (d \multimap j \multimap l) \multimap_* d \multimap_* j \multimap_* l \qquad \llbracket \text{likes} \rrbracket : d \multimap j \multimap l}{A_*(\llbracket \text{also} \rrbracket)(\llbracket \text{likes} \rrbracket) : d \multimap_* j \multimap_* l}
    \qquad \llbracket \text{dogs} \rrbracket : d
  }{A_*(A_*(\llbracket \text{also} \rrbracket)(\llbracket \text{likes} \rrbracket))(\llbracket \text{dogs} \rrbracket) : j \multimap_* l}
  \qquad \boxed{1}
}{
  A_*(A_*(A_*(\llbracket \text{also} \rrbracket)(\llbracket \text{likes} \rrbracket))(\llbracket \text{dogs} \rrbracket))(A_*(A_*(\llbracket \text{comma} \rrbracket)(\llbracket \text{john} \rrbracket))(A(\llbracket \text{who} \rrbracket)(A(\llbracket \text{likes} \rrbracket)(\llbracket \text{cats} \rrbracket)))) : l
}
}{
\langle\langle like(j, \iota x.dog^*(x)), \{like(j, \iota x.cat^*(x))\}\rangle, \{\exists z.like(j,z) \wedge z \neq \iota x.dog^*(x)\}\rangle : l
}
$$

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

7 Conclusion

- The take home messages are as follows:

    1. Multidimensionality is necessary to capture the at-issue/side-issue distinction. Hiding multidimensionality in updates does not help.
    2. The interactions between the two dimensions are restricted — free interaction is not the solution.
    3. This means that we need enough structure to distinguish the different forms of interaction and to limit them.
    4. This additional structure cannot be effectively captured by conjoined terms in the logic of composition.
    5. We have proposed to use monads to simultaneously capture the multidimensionality and to provide enough structure to control interactions between dimensions. In particular we want to restrict the flow of information to a post-compositional phase.
    6. In this way our system avoids the two problems we identified for the Barker et al. (2010) system for the following reasons:
        (a) We have followed Potts's intuition that side-issue content cannot be accessed compositionally and we have modelled this intuition using the *Writer* monad. This automatically partitions the space of possible lexical items properly, such that items like **negex** are correctly precluded. The item **negex** would require a reading interface that the *Writer* monad does not provide.
        (b) Barker et al. do not have a type for the result of the continuation that is common to both quantifiers and side-issue meanings. In our system, quantification is handled by the standard type-logical mechanisms, but side-issue content is handled entirely and independently by monads in the meaning language. The problem therefore fails to arise at all.

- In conclusion, please pick up after your fucking dog.

References

AnderBois, Scott, Adrian Brasoveanu, and Robert Henderson. 2010. Crossing the Appositive/At-issue Meaning Boundary. In Man Li and David Lutz, eds., *Proceedings of SALT 20*, 328–346.

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

Asudeh, Ash. 2004. Resumption as Resource Management. Ph.D. thesis, Stanford University.

—. 2009. Adjacency and Locality: A Constraint-Based Analysis of Complementizer-Adjacent Extraction. In Miriam Butt and Tracy Holloway King, eds., *Proceedings of the LFG09 Conference*. Stanford, CA: CSLI Publications.

—. 2012. *The Logic of Pronominal Resumption*. Oxford: Oxford University Press. To appear.

Barker, Chris, Raffaella Bernardi, and Chung chieh Shan. 2010. Principles of Interdimensional Meaning Interaction. In Man Li and David Lutz, eds., *Proceedings of SALT 20*, 109–127.

Nouwen, Rick. 2007. On Appositives and Dynamic Binding. *Research on Language and Computation* 5(1): 87–102.

Potts, Christopher. 2005. *The Logic of Conventional Implicatures*. Oxford: Oxford University Press.

—. 2007. The Expressive Dimension. *Theoretical Linguistics* 33(2): 165–197.

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

A Monad Transformers

- We need a way to combine monads:
  - In the analysis of (16) we assume the two distinct *Writer* monads.
  - The anaphoric links could be resolved directly during the compositional phase. In order to do so we would need to enrich the monadic meanings to have access to some sort of referents storage (Nouwen 2007). We can use the *State* monad to this effect.
- 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; therefore 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 $M\,x \rightarrow MT\,M\,x$, 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.