Variables without Assignments

Avery D Andrews
ANU, Mar 2011

It often appears to be assumed that the use of variable(-like component)s in linguistic structures requires the use of assignments of values to those variables in order to interpret them semantically in set-based model theory (see Jacobson (2005:186-187) for a recent example). In this note I will show how the use of assignments can be avoided by means of 'indeterminates' in closed cartesian categories (CCCs), as explained in Lambek and Scott (1986:47-65) (henceforth LS). People who know this material need read no further, because the point is actually rather obvious (in spite of the embarrassingly long time it took me to notice it), and remainder of this note is just an attempt to present the basic idea in such a way that people who haven't really learned about CCCs will have some idea of how it works. In particular, I make no claims for any specific empirical advantage for this technique over the use of assignments, but simply point out that it provides an alternative, and neither do I claim any original mathematical content, since the application of LS is simple and immediate.

In spite of this general disclaimer, this approach perhaps has a conceptual advantage over assignments, which is that it works because of the intrinsic properties of the set-based model theoretic interpretations, in particular, that they have enough in common with CCC's to make the Functional Uniqueness Theorem true. This is false for some systems for which assignments can be used, such as some of the intensional logic systems in Muskens (2007) (the ones where alpha-equivalence doesn't hold), and not obviously true (at least to math beginners) for others, such as the one he proposes for linguistic semantics (i.e., FCT would need to be proved for this system, and hasn't been afaik). So I think that the question of whether NL semantics can be done in a system with indeterminates rather than variables is not entirely without interest.

I won't attempt a full presentation of the required category theory here,[1] but only sketch those of the rudiments that are most immediately relevant to making the point vaguely comprehensible.

1 Category Basics

A category consists of two 'collections'[2] obeying some laws. One collection is of 'objects', the other of 'arrows', and the laws are:

[1] For a brief presentation, see Pierce (1991); for a more comprehensive treatment with hundreds of solved exercises, Barr and Wells (1999a). Lawvere and Schanuel (1997) might also be helpful, along with many other introductions to this subject. The basics of universal algebra as presented for example in Burris and Sankappanavar (1981) would be a useful prerequisite. Some other resources include the earlier years of John Armstrong's blog 'The Unapologetic Mathematician', and Eugenia Cheng's and Simon Willerton's 'The Catsters' channel on Youtube.

[2] 'Collection' rather than 'set' in order to avoid commitment to various details about how sets appear to work, especially 'size limitations'.

(1) a. Source/Target: In a category $\mathscr{C}$, any arrow f has exactly one object of $\mathscr{C}$ as its source and one (possibly the same, possibly different) as its target. $f:A \to B$ means that f is an arrow with A as its source and B as its target (deliberately the same as function notation).

b. Composition: if $f:A \to B$ and $g:B \to C$ are arrows of $\mathscr{C}$, then $\mathscr{C}$ has an arrow $gf:A \to C$ (a composition operation symbol such as '$\circ$' is often used, but we'll follow LS and just use juxtaposition).

c. Identities: for any object C of $\mathscr{C}$ here is an arrow $\mathbf{Id}_C:C \to C$ of $\mathscr{C}$ such that, given $f:A \to B$, $\mathbf{Id}_B f = f$ and $f\mathbf{Id}_A = f$

d. Associativity: given $f:A \to B, g:B \to C, h:C \to D$, then $(hg)f = h(fg)$

It might seem a bit fussy to keep mentioning the category $\mathscr{C}$, but a standard category theory technique is to use the arrows of one category as objects of another, and perhaps also some of the arrows from the original category as arrows of the new one, so it can be very important to be explicit about what category something is supposed to be an arrow or object of, since being a category is, mentally, much more a matter of how something is being regarded than one of what it 'really is'.

The only category we'll consider here is called **Sets**, which has ordinary sets as its objects, and functions as its arrows, (e.g. Partee et al. (1993)), with one important technical modification: in set theory, a function is construed as a set of ordered pairs 'from' its domain, the set of elements appearing as first members, 'to' any set that contains its range, the set of elements appearing as second members. But a function-as-arrow must nominate a specific set containing the range to serve as its target. So that we can for example think of multiplication by 2 as a function from the positive integers into the positive integers, or from the positive integers into the rational numbers, etc. But these distinct 'uses' of the set-theoretical function have to be taken as distinct arrows in category theory.[3] This can be implemented by having an arrow $f:A \to B$ in sets consist of an ordered pair $<f, B>$ where $B \subseteq \mathbf{ran}f$.

This example is enough to reveal why we spoke of 'collections' rather that 'sets' of objects and arrows: there can't be any set of all sets, due to Russell's Paradox, but without a 'collection' of all sets (perhaps some kind of 'proper class'), we couldn't have a category of sets. On the other hand, granted the collection of objects, we are in a somewhat better position for arrows, since, given any two sets A, B, standard set theory provides for a set of arrows from A to B (standardly notated as B^A). This makes **Sets** a 'locally small' category.

[3]Which carries some of the appeal of category-theory for computer science, since the implementation of multiplication by 2 for integers is likely to be quite different from that for rationals or the 'floats' that approximate reals.

2 Arrows for Members

A basic theme of category theory is not to think at all about the internal structure of objects, but only in terms of the structure of the relationships between them, as encoded in the arrows. This creates an immediate problem as to what to do about the set-theoretical notion of 'member', since it seems rather fundamental to set theory that sets consist of members, which category theory wants to discourage us from thinking about.

But there is a trick that comes to the rescue, based on the fact that if we take any one-member set as a given, traditionally notated as $\{*\}$, the traditional members of any other set A are in one-to-one correspondence with arrows from $\{*\}$ to A. Such arrows are called 'elements'; we will be rigorous about only using the term 'member' for traditional set-theoretical members, and 'element (of A)' for arrows from $\{*\}$ to A.

But of course we haven't broken free from the notion of 'membership' if our concept of 'element' depends on the idea of a set with one member, so it is fortunate that there's a purely arrow-based version of that idea too, 'terminal object':

(2) An object T of a category $\mathscr{C}$ is a terminal object iff, for any object A of $\mathscr{C}$, there is a unique arrow of $\mathscr{C}$ from A to T, which we'll notate as $\bigcirc_A$,

'Unique' here meaning given any putative arrow $f{:}A \to T$, it can be proved that $f = \bigcirc_A$ (no use of a membership concept required). $\{*\}$ is clearly a terminal object (by extensionality of functions). Terminal objects (lots of them) provably exist in **Sets**, and are the easiest-to-understand requirement for a category to be a CCC, where the symbol 1 is the commonest choice for the terminal object.

So now we can define an 'membership-free' notion of element as follows:

(3) In category with a terminal object 1, an element of an object A is an arrow $f{:}1 \to A$.

3 Products

The next apparatus we need for a CCC is called 'products', including a 'product object' and various accessory arrows, obeying some laws. The usual choice of product object for **Sets** is the regular cartesian product: given any two objects/sets A, B, we have the product object/set $A \times B$. One major piece of the apparatus is the 'pairing' operation on arrows: given arrows $f{:}C \to A$ and $g{:}C \to B$, it is required that there be a pairing arrow $<f, g>{:}C \to A \times B$. Standardly for **Sets**, this will be the function $c \mapsto <f(c), g(c)>$.

This might be expected to induce a clash with the regular ordered pair notation for set theory, but actually, it doesn't. For in place of members a, b, we'll have elements $a{:}1 \to A$, $b{:}1 \to B$. But then $<a, b>$ will be an arrow from 1 to $A \times B$, and, given the various principles that we haven't gone thru here, it will be the correct one, namely, the

arrow that goes from $\{*\}$ to the conventional ordered pair $<a(*), b(*)>$. So everything works out nicely.

A category with a terminal object, products, and the associated accessories all obeying the laws properly is called a 'Cartesian Category' (CC).

4 Exponentials and CCCs

The final piece of apparatus for a CCC is called an 'exponential', and in *Sets*, the exponential object 'from' A 'to' B is the function-space B^A (happily, we turn out not to have to worry about explicitly packing the members/elements of B^A with their target B). Amongst the bits and pieces that comes with it, all we need here is the 'evaluation arrow from A to B', $\mathbf{ev}_{A,B}: B^A \times A \to B$, which takes a pair $<f, a>$, where f is a member of B^A, and A is a member of A, and maps this pair to the member $f(a)$ of B. This is enforced by laws involving arrows, which provide as a by-product for the 'currying' of functions of two-variables (i.e. whose domains are products).[4]

We now enough to set up a semantics for function-evaluation expressions. Traditionally, a proper name such as *Uther* might be modelled by a member of some set E (for 'entities'; 'A' seems to occur more often in semantics textbooks), while intransitive verb such as *sleep* would be modelled as a member of B^E, B for 'boolean value', i.e. something like $\{0, 1\}$, where '1' is taken as meaning 'truth' (or maybe 'P' for 'proposition', if we want to take on (hyper)intensionality). In our category theoretic remodelling, these two 'members' will be replaced by elements, i.e. arrows from 1. Then, to evaluate an expression such as *Sleeps*(*Uther*), what we do is make a pair and compose $\mathbf{ev}_{E,B}$ to it:

(4) $\mathbf{ev}_{E,B}<sleep, Uther>$

The result of composing these arrows will be an element of B, i.e. a truth-value, as desired (or some more complicated kind of thing, such as a 'proposition', in an intensional or hyperintensional model).

Dealing with curried functions as meanings for transitive verbs is done with multiple evaluation arrows of the appropriate types. An interpretation for '*love*' might for example be an element of B^{E^E} (other notations for the exponential get more convenient here, such as $[E \Rightarrow E \Rightarrow B]$ (implicit association to the right)), with the interpretation of *loves*(*Gwen*)(*Lance*) (implicit association to the left) being:

(5) $\mathbf{ev}_{E,B}<\mathbf{ev}_{E,E\Rightarrow B}<love, Gwen>, Lance>$

It is then convenient to informally notate (5) as *love*(*Gwen*)(*Lance*), which could be intepreted as a formula in a formal language, but can also be viewed as informal notation of (5).

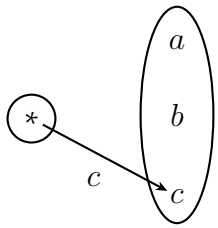
[4] LS is on the whole a bit crisp to serve as an introduction to category theory, but the working through of this part of the material is as clear as any other that I've seen.

Replacing members with elements has set us up for indeterminates, to which we procede now.

5 Indeterminates

The idea of adjoining indeterminates to a CCC is as follows. An element of a set A in the category **Sets** is an arrow from 1 to A. Traditionally, one might represent it like this:

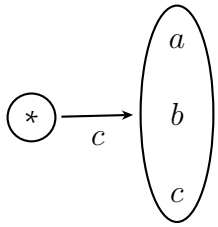
(6) 1 A



Here the graphical arrow, labelled 'c', goes from $*$ to the specific member of A, also designated 'c', that the element c maps $*$ to.

But since we're not supposed to be interested in internal structure, amore 'categorically correct' picture would be:

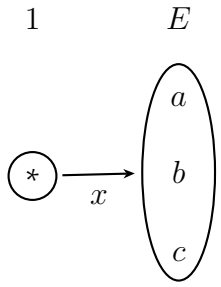
(7) 1 A



The point being that, considered categorically, functions, including elements, are just a kind of 'blob-connector'.

This leads to the following thought: Suppose we start with **Sets** (or, more modestly, just that corner of **Sets** where are models live, E, B and the objects and arrows we can make with the elements of these sets and the CCC apparatus), and just add to it some additional arrows from 1 to E (or any other of our objects). Such an 'indeterminate' arrow can be diagrammatically represented like this:

(8)

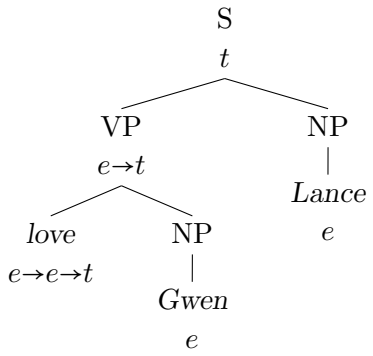


Of course *a priori* we can't just assume this is possible, but LS prove that it can be done, yielding from a CCC $\mathscr{C}$ a new, uniquely determined (up to 'isomorphism') CCC that we can notate as $\mathscr{C}[x]$.[5]

The situation is highly analogous to taking a ring R and adding to it a variable x (also sometimes called an 'indeterminate' iirc) to get the ring of polynomials in x, and various things are provable in both situations, such as that the results of 'adjoining' different indeterminates to the same category are isomorphic ($\mathscr{C}[x] \cong \mathscr{C}[y]$), and that adjunction of indeterminates can be iterated to get $\mathscr{C}[x][y]$ ($\cong \mathscr{C}[y][x]$).

To use indeterminates to do the work of variables, we will need some way to 'eliminate; them (equivalent to binding a variable), which is achieved with the 'Functional Uniqueness Theorem', but first we need to tweak evaluation so as to deal with them. Indeterminates are going to serve as interpretations for syntactic 'variables', with the effect that using them is going to change the category in which we look for an interpretation. Treating our formulas such as *loves*(*Gwen*)(*Lance*) as expressions in a language with binary-brancing structure:

(9)



We can say that the interpretations of the terminal nodes are found in **Sets**, and likewise for the phrases, by our equivalent of function application.

But suppose we use indeterminates as interpretations for items such as 'traces' or bound pronouns, perhaps to get a semantics for relative clauses such as *who Lance*

[5]Indeed, they work though three ways of doing it, two of them 'syntactic' in flavor, making use of the free category construction, the other 'algebraic', via the co-Kleisli category construction. And an additional 'algebraic' construction is pointed out in the exercises. 'Algebraic' meaning that nothing based on an expression in any language is involved.

loves. We can say that the putatively empty/'trace' NP in object position gets for its interpretation the indeterminate x, but now the interpretation for *loves e* will be $\mathbf{ev}_{E,E\Rightarrow B}\langle love, x\rangle$, which is indeed an arrow from 1 to $E\Rightarrow B$, but not one that lies in **Sets**, but rather **Sets**$[x]$. And then when we combine this with the remaining argument, we might get an arrow in an even bigger category (if there's a bound pronoun in the subject, etc).

So, at each stage of composition, we need to note which indeterminates are needed in the category of interpretation, when we combine two interpretations, to pool the indeterminates. Union would seem the appropriate operation for this, since for a relative clause such as *who thinks Gwen loves him*, the subject and object of the complement clause would presumably use the same indeterminate. This is highly comparable to keeping track of the variables for which values must be provided when 'finitistic' assignments are used, as in Kratzer and Heim (1998), and also in Lawvere's interpretation of first-order quantifers as adjoints.[6]

6 Functional Completeness and Elimination

Adding indeterminates won't be useful if we can't get rid of them, so as to get interpretations back in **Sets**, where we want our models to live, so it is fortunate that they can be removed by virtue of the 'Functional Completeness Theorem' (FCT). Suppose $\phi(x)$ is an arrow in $\mathscr{C}[x]$, that is, a combination of arrows from $\mathscr{C}$, together with x (which might appear in one place, many, or none, in the combination; '(x)' in '$\phi(x)$' is just an allusion the fact that the arrow might involve x). What this theorem says is that for any such arrow $\phi(x){:}1 \to B$, where $x{:}1 \to A$, there is a unique arrow $h{:}1 \to B^A$ of $\mathscr{C}$ such that $\mathbf{ev}_{A,B}\langle h, x\rangle = \phi(x)$. If we think of the x as a sort of 'contamination' that is present in some of the arrows of $\mathscr{C}[x]$, the theorem says that we can produce a 'pure' arrow of $\mathscr{C}$ that can be combined in a simple way with a single instance of x to produce the same result as the original $\phi(x)$. And because this arrow h is the unique one with the required property, we can denote it with a term based on $\phi(x)$ and x, for which LS's is $\lambda_{x\in A}\phi(x)$ (the membership notation is used to specify the type of the indeterminate).

One might think that this notation is an introduction of standard lambda-calculus, and in some respects that is the case (more on this shortly), but not exactly, because it doesn't have to be taken as part of any formal language with a definite relationship to an model-theoretic interpretation, but as in informal mathematical notation for a mathematical object that is guaranteed by a theorem to exist in the original semantics category.

Unsurprisingly, the techniques used to prove the theorem are essentially the same as those used in variable free semantics, or to define lambda-abstraction in combinatory logic, but a difference here is the fact that since the arrow so-defined is unique, we can

[6]Discussed lucidly by Todd Trimble here: `http://golem.ph.utexas.edu/category/2007/10/concrete_groups_and_axiomatic.html`

write a term to designate it without regard to any particular story about how it is to be produced, since we know that it always can be.

A relative clause construction can then be interpreted by means of the FCT in the obvious way, to get a meaning lying in a category with one less indeterminate. Imitating Kratzer and Heim's (1998:96) rule of 'predicate abstraction', we can write:

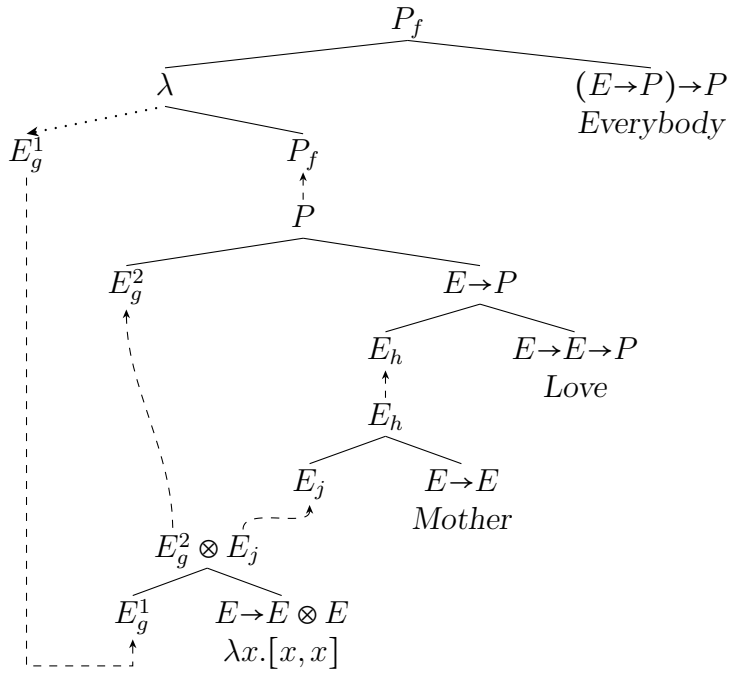
(10) If α is a branching node whose daughters are a relative pronoun indexed with indeterminate x of type A, and β with interpretation in $\mathscr{C}[x]$, then $[[\alpha]] = \lambda_{x \in A}[[\beta]]$, with interpretation in $\mathscr{C}$.

It would be nice to have a way of redoing this to eliminate the syncategorematicity, but I don't know how to do that now.

7 Application to Glue Semantics

For a more systematic application, we can use these techniques to provide interpretations in any CCC to LFG's 'glue semantics', essentially by extending Perrier's (1999) 'maximal labelling' of proof nets to apply to tensors as well as implications. This requires glue to be packaged as in Andrews (2010a) as propositional linear logic rather than system F as usual, or first order linear logic (Kokkonidis 2008). As discussed in the Andrews paper, and in a more relaxed manner in Andrews (2009), we first reorganize proof-nets so that the net for a sentence such as *Everybody loves their mother* looks like this:

(11)



As discussed in Andrews (2009, 2010a), the dashed arrows represent axiom links, while the solid lines represent the 'dynamic graph' of de Groote (1999), oriented upwards,

and the dotted lines connect the 'positive antecedents' to their 'negative antecedents' as in conventional proof-nets.

We begin by assigning intepretations to the nodes with semantic labels, and the negative antecedents, as follows:

(12) a. The interpretation of a node with a semantic label is an element of the semantic type of that node, in the CCC in which interpretations are being found.

b. The interpretation of the negative antecedent of a positive implication is a unique indeterminate of the type of that antecedent.

Some semantic labels are arbitrary constants, e.g. $Love{:}1 \rightarrow E{\Rightarrow}E{\Rightarrow}P$, while others are specifiable in terms of the CCC ingredients, such as the pronominal meaning $\lambda x.[x, x]$, expressible in LS' 'internal language' for a CCC as $\lambda_{x \epsilon E} <x, x>$ (the notation used in the tree follows LFG glue conventions in not specifying the type of the variable).

These values propagate through the net in accordance with the following rules:

(13) a. The semantic value at the source of an axiom link propogates to its target (we regard the source as the positive polarity node of the two that are linked).

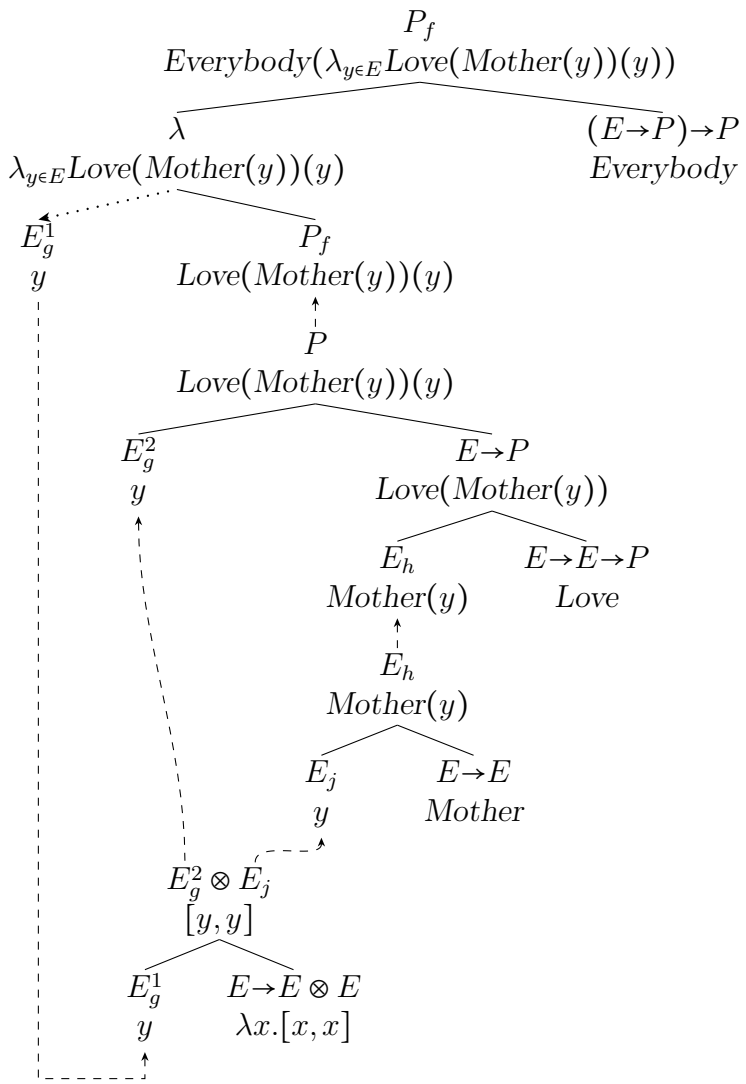
b. The semantic value of the mother of a negative implicatation (such a mother is one with two daughters connected by solid lines, one an implication, the other its antecedent) is the value of the implication applied as a function to the value of the antecedent as argument.

c. The semantic value of the left and right components of a negative tensor (positive tensors aren't used here) are the left and right projections of the value of the tensor itself.

d. The semantic value of a positive implication (a node with one dotted-line daughter, its antecedent, and one solid line daughter, its consequent), is the unique arrow determined by using the FTC to eliminate the indeterminate assigned to the antecedent.

The result of annotating (11) according the these rules is:

(14)



These labels designate arrows in a CCC, but we need to indicate exactly what CCC each arrow lies in.

For the nodes with semantic labels, this category is the CCC $\mathscr{C}$ in which the interpretations are to be found, while in a positive antecedent labelled with the indeterminate x, this category is $\mathscr{C}[x]$. For the nodes whose semantic values are determined by propagation according to the rules of (13), the categories are determined as follows:

(15) a. If the value at the source of an axiom link lies in $\mathscr{C}[X]$, so does that at the source, where $\mathscr{C}[X]$ is $\mathscr{C}$ with the indeterminates belonging to the set X of indeterminates adjoined (some order of of adjunction must be chosen, but which is arbitrary, and X can of course be empty).

b. If the value of a negative implication lies in $\mathscr{C}[X]$, and its sister/antecedent in $\mathscr{C}[Y]$, then the value of the consequent/mother of the implication lies in $\mathscr{C}]X \cup Y]$.

c. If the value of a negative tensor lies in $\mathscr{C}[X]$, then so do the values of both components.

d. If the value of the consequent (sold line/right daughter) of a positive implication lies in $\mathscr{C}[X]$, and its antecedent (dotted line/left daughter is labelled with x (alternatively, lies in $\mathscr{C}[x]$), then the value of the implication lies in $\mathscr{C}[X - x]$.

These rules might seem rather arbitrary and artificial, but Andrews (2010b) shows how to treat them as a technique for computing a semantic value that is determined in a less arbitrary way.

8 General Consequences

Is there any payoff for replacing assigments with this kind of apparatus? Not necessarily. Indeterminants are after all a kind of 'artificial ingredient' in the model theoretic objects, that might be deemed harmful, and seem to be entirely dispensed with in Jacobson's variable-free semantic program. On the other hand this is not free of troubles (for example depending overmuch on the ungrammaticality of weak crossover violations to justify the $z-$ operation of Jacobson (1999)), and this technique for dealing with variables is clearly much less artificial than the full apparatus of assignments. One indication of this is that it doesn't work unless the semantic interpretations constitute a category supporting an FCT, which, for example, the interpretations constructed in the earlier parts of Muskens (2007) do not, since they don't support alpha-equivalence. Another potential problem is Brasoveanu's proposals (`http://people.ucsc.edu/~abrsvn/P_n_P_Sem.html`) to solve various problems in the interpretation of pronouns with antecedents involved in quantification by treating assignments as 'first class citizens' in the interpretation. Purging them might well leave us with no adequate account of these problems.

For now, I will simply take this as evidence that this approach to dealing with variables at least manages to clamber over the 'not even wrong' hurdle, leaving the issue of whether it's actually right for the future.

References

Andrews, A. D. 2009. Yet another attempt to explain glue. `http://arts.anu.edu.au/linguistics/People/AveryAndrews/Papers`.

Andrews, A. D. 2010a. Propositional glue and the correspondence architecture of LFG. *Linguistics and Philosophy* 33:141–170.

Andrews, A. D. 2010b. Semantic interpretation by cc functors for lfg+glue. URL: `http://semanticsarchive.net/Archive/mIzODhmY/`, or `http://arts.anu.edu.au/linguistics/People/AveryAndrews/Papers`.

Barr, M., and C. Wells. 1999a. *Category Theory for Computing Science*. Montréal: Centre de Recherches Mathématiques. 3rd edition. can be ordered from: `http://crm.umontreal.ca/pub/Ventes/CatalogueEng.html`. The most essential content for linguists, but no exercises, can be found in Barr and Wells (1999b), available free online.

Barr, M., and C. Wells. 1999b. Category theory lecture notes for ESSLLI. URL: `http://www.let.uu.nl/esslli/Courses/barr-wells.html`. a condensed version, without exercises, of Barr and Wells (1999a). It is impressively well targeted on material that seems likely to be useful to linguists.

Burris, S., and H. Sankappanavar. 1981. *A Course in Universal Algebra*. Springer Verlag. updated edition available online (Nov 2004) at `http://www.thoralf.uwaterloo.ca/htdocs/ualg.html`.

de Groote, P. 1999. An algebraic correctness criterion for intuitionistic multiplicative proof-nets. *Theoretical Computer Science* 224:115–134. Retrieved 15 November, 2010, from `http://www.loria.fr/~degroote/bibliography.html`.

Jacobson, P. 1999. Towards a variable-free semantics. *Linguistics and Philosophy* 22:117–184.

Jacobson, P. 2005. Direct compositionality and variable-free semantics: the case of 'Principle B' effects. In *Direct Compositionality*. Oxford University Press.

Kokkonidis, M. 2008. First order glue. *Journal of Logic, Language and Information* 17:43–68. First distributed 2006; URL: `http://citeseer.ist.psu.edu/kokkonidis06firstorder.html`.

Kratzer, A., and I. Heim. 1998. *Semantics in Generative Grammar*. Oxford: Oxford University Press.

Lambek, J., and P. J. Scott. 1986. *Introduction to Higher Order Categorical Logic*. Cambridge University Press.

Lawvere, F. W., and S. H. Schanuel. 1997. *Conceptual Mathematics : A First Introduction to Categories*. Cambridge University Press.

Muskens, R. 2007. Intensional models for the theory of types. *Journal of Symbolic Logic* 72:98–118.

Partee, B. H., A. ter Meulen, and R. E. Wall. 1993. *Mathematical Methods in Linguistics*. Kluwer. Corrected second printing.

Perrier, G. 1999. Labelled proof-nets for the syntax and semantics of natural languages. *L.G. of the IGPL* 7:629–655.

Pierce, B. 1991. *Basic Category Theory for Computer Scientists*. MIT Press.