Direct Compositionality and Variable-Free Semantics: The Case of Binding into Heads*

Pauline Jacobson
Brown University

1. Theoretical Background

The point of departure for this paper is the hypothesis of "direct compositionality" (see, e.g., Montague 1974): the syntax and the model-theoretic semantics work in tandem. Thus the syntax "builds" (i.e. proves the well-formedness of) expressions, where each syntactic rule supplies the proof of the well-formedness of an output expression in terms of the well-formedness of one or more input expressions. (These rules might, of course, be stated in highly general and schematic terms as in, e.g., Categorial Grammar.) The semantics works in tandem with this - each output expression is directly assigned a meaning (a model-theoretic interpretation) in terms of the meaning(s) of the input expressions(s). There is thus no need to for any kind of abstract level like LF mediating between the surface syntax and the model-theoretic interpretation, and hence no need for an additional set of rules mapping one "level" of syntactic representation into another.

Within this general program, I will be advocating a sub-program: that of Variable-Free semantics (see, e.g., Jacobson, 1994, 1999; 2000). The claim here is that the semantics makes no use of variables and hence no use of assignment functions. The relevance of this to the hypothesis of direct compositionality stems from the observation that many of the arguments for LF are based on the notion that "binding" is a relationship between an actual "binder" (an NP, or DP) and a pronoun, trace, and/or variable at some level of representation. It is further assumed that the "binder" and the "bindee" have to be in a particular syntactic configuration with respect to each other at the relevant level. But because we frequently find "binders" and "bindees" which are not in the right configurational relationship on the surface (or at least not at the "pronounced"part of surface structure), it is common to posit an abstract level (or unpronounced pieces) at which they *are* in the right configurational relationship. The variable-free program maintains that this entire view is incorrect. The claim here is that there is no such thing as "binders" and "bindees" - the semantic effect that we think of as "binding" is the result of a rule which merges *argument slots*. Hence "binding" is not a relationship between two actual NPs in some syntactic representation.

This paper is an in-depth study of one such case - the case of the binding of a pronoun in head position from a "binder" within a relative clause. . On the one hand, we will see (Sec. 4) that positing an abstract level to account for the "binding" here is actually of no help. Among other problems, it does not get the semantics right (at least not in any straightforward way). On the other hand, we will see that assigning the correct meaning is straightforward under the variable-free program. To do this, I will propose one new type shift rule above and beyond the apparatus developed in Jacobson (1999), but I will argue that this particular rule is extremely simple and natural. There is an additional point of interest emerging from this domain: it provides an illuminating comparison between a theory with and without variables. Thus something akin to the proposal here could be implemented in a theory with variables, and in fact my proposed solution is arguably just a variant of the solution in Engdahl (1986) for this type of problem. Engdahl too adopts a version of direct compositionality, but implements this using variables. Because the two solutions vary essentially in whether or not they adopt variables and

B. Jackson (ed), *SALT XII* 144-163, Ithaca, NY: Cornell University.

assignment functions, this domain highlights a key difference between variable-free and variable-ful. Thus I will argue that Engdahl's version lacks the "naturalness" and simplicity allowed in the variable-free account - this is precisely because of the clutter engendered by assignment functions. As soon as one recasts her basic insight into the variable-free way of thinking, it becomes perfectly natural.

2. Background Pieces

Consider the semantic composition of relative clauses in general, as in:

(1) the cat that Mitka chased

I make some minimal and fairly standard assumptions here. One is that the meaning of *(that) Mitka chased* is $\lambda x[\text{chased}'(x)(m)]$. I further assume that the grammar contains some rule(s) which allow *cat* to combine with *that Mitka chased* to give a complex noun (which then occurs with the determiner), whereby the meaning of *cat that Mitka chased* is the intersection of cat' with that-Mitka-chased'. (I use the prime notation here to notate model-theoretic objects - not objects in some intermediate representation. I will, moreover, be referring to complex material *cat that Mitka chased* as a "noun" and will assign it the category N. One can substitute one's favorite category label here..) Further, take a case with two relative clauses like:

(2) the cat that Mitka chased that lives next door

I will assume a standard "stacking" analysis here - whereby *cat that Mitka chased* first combines to give a complex head which then combines with the second relative clause. (Hence cat' intersects with that-Mitka-chased' which result in turn intersects with that-lives-next-door'. See Sec. 4.3 for an alternative possibility.

The next background piece is less standard, and concerns the question of how binding takes place into the post-copular constituent in (3) (such cases were first noted in Geach (1972) and have played a prominent role in recent literature):

(3) a. The woman who every man$_i$ loves (the most) is his$_i$ mother.
b. The (only) woman that no man$_i$ would forget to invite to his wedding is his$_i$ mother.

Incidentally, the use of indices here and throughout is merely for convenience to indicate the intended reading - in the variable-free view the grammar knows nothing about indices. Here I assume the general line taken in von Stechow (1990), Jacobson (1994) and Sharvit (1998) - who all observe that these are a straightforward extension of functional questions as analyzed in Groenedijk and Stokhof (1983) and Engdahl (1986). This basic observation is neutral between the standard and the variable-free views (note, for example, that von Stechow and Sharvit's versions of the analysis are embedded in a standard framework). I have, however, argued elsewhere that the analysis functional phenomena (question and NPs) in general is much simpler and more natural in a variable-free framework. In fact, the existence of functional readings for questions and NPs is completely unsurprising in the variable-free view and so these readings come virtually for free (see Jacobson (1999) for discussion). I will, therefore, elucidate the functional analysis of (3) using the variable-free apparatus.

To save space, I will have to assume some basic familiarity with the apparatus discussed in Jacobson (1999), but will briefly review its applicability to

the case at hand. (This is detailed further in Jacobson, 1994.) Take first a relative clause such as *(who) every man loves.* In an ordinary case, this is of type <e,t> - and denotes the set of individuals that every man loves. This result can be derived rather straightforwardly (and in a direct compositional fashion) under, e.g., the analysis of extraction in Steedman (1987) or under any of the other extraction analyses which have been put forth in the Categorial Grammar and related literature. But given the variable-free apparatus, this relative clause also automatically has an additional "functional" meaning, by which I mean a meaning of type <<e,e>,t> (i.e., a set of functions from individuals to individuals). We need nothing new in the system to derive this: the very apparatus used for pronominal "binding" in general automatically gives us this meaning. Thus *loves* can undergo the **z** rule to be of type <<e,e>,<e,t>> with meaning $\underline{\lambda f[\lambda x[loves'(f(x))(x)]]}$. The normal conventions will apply to put together the full relative clause, and - under the derivation where *loves* has undergone **z** - the meaning will be $\underline{\lambda f[every\text{-}man'(\lambda y[loves'(f(y))(y)])]}$. (In other words, this is the set of functions that every man **z**-loves.) This, then, is what happens in (3a). The same basic point holds for the case of functional questions. Notice that functional readings for questions are somewhat unexpected under the standard view of things - the usual line is to assume that a "trace" or "gap" can correspond either to an individual variable or to a variable over functions of type <e,e> applied to a variable over individuals (see, e.g., Groenendijk and Stokhof (1983), Engdahl (1986)). But it follows from nothing else that a gap should be able to have either meaning. In the variable-free system, a "gap" is nothing more than an argument slot which hasn't been filled in the "normal" way. Moreover, the **z** rule is the type-shift rule which accomplishes binding *in general.* (Notice that any theory needs *some* rule to accomplish binding.) Thus the very mechanism for binding in general can apply here so as to give a functional "gap" rather than an individual gap.

The next step in the semantic composition of (3a) is the only piece which does not come for free in the variable-free program. Thus, we need a type-shift rule to map the meaning of the head noun *woman* (whose lexical meaning is a set of individuals) into a set of functions whose range is $\underline{woman'}$. This is given in (4);, I leave it open as to the effect this rule has on the syntactic category of the expression:

(4) Let α be an expression of the form: $<[\alpha];\ N;\ \alpha'>$. Then there is an expression β of the form $<[\alpha];\ N\ (??);\ \lambda f[\forall x_{\text{in domain of f}}]\ \alpha'(f(x))]>$ for f a (partial) function of type <e,a>, where $\alpha'$is of type <a,t>.

Two comments are in order . First, in most cases of concern f is of type <e,e>, as the input to the rule is a noun with meaning of type <e,t>. However, I have formulated (4) more generally and hence this can apply recursively. This will, for example, allow a noun to shift to have a meaning of type <<e,e>,t>, and then to shift again into a meaning of type <<e,<e,e>,t> etc. (It should perhaps be given even more generally so that the output can be of any type <b,a>, but we will not pursue this here.) Second, one does not want a proliferation of type-shift rules, and it would be nice if (4) followed from something else. As of yet, though, I do not see how to eliminate this (but see Winter this volume). But in defense of (4), it seems like a rather natural and unpernicious rule. And note that the non-variable free analogue to the analysis here *also* makes use of this shift (see, e.g., von Stechow (1990), Sharvit (1998)) and so this is not extra to the variable-free analysis.

The final piece in the analysis does comes "for free" in the variable-free program: the apparatus is such that, in general, an expression like *his mother* denotes a function from individuals to individuals (rather than the standard view under which it is a function from assignment functions from individuals). We now

have all the pieces to give a smooth, simple, direct compositional analysis for (3a) without reconstruction and without use of any additional apparatus except for the rule in (4). *woman* denotes a set of functions with range woman' and *every man loves* denotes the set of functions f such that every-man **z**-loves f. These two intersect - as in the case of combining any head with a relative clause - and this then occurs as argument of *the*. Hence the pre-copular constituent denotes the unique (contextually salient) function f with range woman' such that every man **z**-loves f. The meaning of the post-copular constituent *his mother* automatically is a function of the right type and in particular is the-mother-of function, and *be* equates these.

The moral, then, is that the apparent "binding" of *his* is an illusion. There is no need to posit any kind of extra level/structure at which *his* is c-commanded by *every man*, nor is there any need for any kind of co-indexation between *every man* and the pronoun. Devices like "binding" and indices are artifacts: the semantics just smoothly works to get the appropriate meaning here.

3. The problem: "binding" into the head

That's the good news, but unfortunately not the end of the story. For the appearance of binding is possible not only in the post-copular constituent but also in the head (both in a functional question and in a functional NP) and the story told above says nothing about these:

(5) Which of his_i relatives does every man_i love the most? (His mother)
(6) a. The relative of his_i that every man_i loves (the most) is his_i mother.
b. The relative of his_i that no man_i would forget to invite to his wedding is his_i mother.

Cases like (5) are discussed extensively in Engdahl (1986), whose analysis I return to later. Notice that in the above examples we get apparent "binding" into the complement of a head noun. Unsurprisingly, there is also binding into a relative clause within a complex head or fronted *wh* phrase:

(7) Which woman that he_i loves did every man_i invite?
(8) a. The woman that he_i loves that every man_i invited was his_i mother.
b. The woman that he_i loves that no man_i will forget to invite is his_i mother.

Before continuing, let us consider just where we get this kind of apparent binding. Jacobson (1999) argues that NPs can have functional readings in general (not just in copular constituents), and that allowing for a functional reading on the object NP in, e.g., (9a) explains the presence of the unexpected - or "sloppy"- inference shown in (9c) and discussed in, among others, Reinhart (1990):

(9) a. John always buys whatever Bill buys.
b. $Bill_i$ buys his_i favorite car.
c. Therefore, $John_j$ buys his_j favorite car.

Leaving aside the full details, (9a) means that John **z**-buys whatever function f Bill **z**-buys. (9b) says that Bill **z**-buys the function mapping each individual into that person's favorite car. Again all of this is automatic: it follows immediately that *buy* can undergo **z** in all of its occurrences in (a) and (b),and it follows immediately that *his favorite car* denotes a function of type <e,e>. From this (c) follows - John **z**-buys the favorite car function also.

It is therefore unsurprising that the unexpected "binding" of pronouns in the head occurs in these functional NPs as well:

(10) John forgot to invite the very relative of his that no other man would have ever forgotten to invite (namely, his mother).

The account to be developed below will straightforwardly extend to this case, as the interested reader will easily be able to verify. As will become clear, my account is tied in to the fact that we are dealing in all of these cases with *functional* NPs.

But there is one complication which I will not address: surprising binding effects are occasionally found in other places. Cases along the lines of (11) have been noted from time to time in the literature; here the NP in question does not obviously have a functional reading (or, if it does, then one needs further conventions to put the meanings all together):

(11) The relative of his_i that every man_i loves the most took his_i picture at the family reunion.

These seem to be a special case of a more general phenomenon discussed in Sharvit (1999), who notes further that the phenomena here is far more restricted than the apparent "binding" into functional NPs. (Sharvit deals primarily with Hebrew and concentrates on the apparent "binding" of the pronoun in object position rather than the pronoun within the head, but her observations extend directly tho this case.) For example, these are not terribly happy with downward entailing binders:

(12) ?*The (very) relative of his_i that no man_i would forget to invite always pays for his_i wedding (namely, his_i mother).

.

Sharvit therefore analyzes this type of case as involving a "pair/list" relative - I will not explore whether the account of binding into heads that I propose here will extend to these cases.

Returning to the better understood functional cases, our task is thus to account for apparent "binding" into the head position in (6) and (8). Before developing my proposal, let me mention three other accounts. The first is that in Engdahl (1986), to which I will return in Sec. 7. The second is an account developed in Winter (this volume). This turns out to be in the same spirit as the account here and our accounts (developed independently) have a striking affinity: they are, in fact simply inverses of each other. I will comment on this below. But the approach which is probably best known is a "reconstruction" analysis - according to which there is a level of representation at which the *wh* phrase in questions and the head of a relative clause is in the position of the gap. Since this is so widely assumed to provide a solution to the puzzle, let me digress to look more closely at this approach.

4. Interlude: Deconstructing reconstruction

4.1. The incorrect semantics

Under a "reconstruction" story, the idea regarding (5)and (7) (the question cases) is to posit a level of representation at which the fronted *wh* material is in the position of the gap, and to assume that the constraints on binding (and/or the interpretation of the sentence) holds for that level. Since these are *wh* questions, this goes hand-in-hand with the standard assumption that the fronted *wh* constituent starts out in the gap position and then moves. Hence the level relevant for binding could be the pre-movement level (the hypothesis that interpretation happens *before* movement was standard in the classical transformational and

Generative Semantics literature of the 60' s and 70's); it could instead be a "reconstruction" level at which material is moved and then later put back into the position from which it is moved (as in standard GB); or it could instead be thought of as the actual surface structure under the copy theory of movement (whereby moved material remains in its pre-movement position, albeit in "unpronounced" form). I will henceforth use the term *reconstruction solution* to refer to any of these three views. The explanation for the appearance of binding in a relative clause head as in (6) or (8) is somewhat trickier - the analogous story can be maintained only with the additional assumption that the *head* in a relative clause (as well as the *wh* - word) starts out in the gap position. This view - the Head Raising analysis - is considerably more controversial. But it has been proposed for this general kind of problem off and on since at least as early as Vergnaud (1974). Thus in (6) the head is *relative of his* and so - under a Head Raising analysis - is in the gap position at the relevant level. In (8x) (assuming a stacking analysis) the head is *woman who he loves* and so this is object of *invite* at the relevant level. (Of course in this case *woman* is also raised from the gap position following *loves*.)

To the extent that there are specific proposals for what it means to interpret the head noun in the position of the gap, I believe that the usual current view is that the lexical content of the head restricts the variable to range over members of the set denoted by the head. In other words, imagine that the "reconstruction" structure for an ordinary relative clause like (13) is as shown in (14a) or (14b). (The choice here depends on whether or not one adopts a copy theory of movement - in which case *man* is in two places - or whether the head is actually moved into (or, out of) the gap position. In either case, though, it will not (or need not) be interpreted in head position, so in (14b) I will put the head in italics, adopting a convention to italicize material which is "invisible" to the semantics).

(13) every man who Mary invited

(14) a. every [who Mary invited $t_{i\,[man]}$] b. every *man* [who Mary invited $t_{i\,[man]}$]

Assume further the following convention for interpreting the trace:

(15) $[[t_{i[N]}]]^g = g(x_i)$ if $g(x_i) \in [[N]]^g$ and undefined otherwise

Given this. *man who Mary invited* in (13) ends up with the same meaning as in the standard view (according to which the semantic composition intersects the man-set with the set of Mary invitees). That is, $[[\text{Mary invited } t_{i\,[man]}]]^g$ is true if Mary invited $g(x_i)$ and $g(x_i)$ is a man. λ-abstracting over the variable in object position will yield something which has the same value on all assignment functions, and in particular will yield (for every assignment function) a function mapping an individual a to true iff Mary invited a and a is a man.

But for the case of an NP like *the relative of his that every man invited* this procedure yields the wrong meaning. The interested reader can work through the details, but in this case *relative of his that every man invited* will denote the set {x|every-man'(λy[x is a relative of y & y invited x]}. *the* maps this into the unique (contextually salient) member of this set and so we get the unique (contextually salient) individual who is a relative of every man and was invited by every man. This, of course, is of no use in (6a) - what we need here is a functional meaning. In other words, this analysis misses the point that this unexpected binding goes hand-in-hand with the fact that these NPs have functional interpretations (modulo the open cases such as (11) which have a far more limited set of binders).

Now one might try to save the recsontruction analysis by adopting reconstruction and combining it with a functional NP analysis. Consider the

ordinary functional case like (3a) *The woman who every man loves is his mother.* Combining head reconstruction with a straightforward extension of the Groenendijk and Stokhof/Engdahl analyses of functional questions, one would simply say the following: (i) the trace here is complex (let us index it as $t_{f(i)}$; (ii) this translates as a variable over functions of type <e,e> applied to a variable over individuals - hence, as $\underline{f}(x_i)$ (note: the $\underline{f}$ variable should also be indexed but I ignore that here); (iii) *woman* is in the gap position, and in this case (iv) the role of the head noun is to restrict the range of the function. Thus the structure for the functional reading of this NP is (16), and the interpretation of the functional trace is as in (17):

(16) every man loves $t_{f(x)\ \text{woman}}$
(17) $[[t_{f(i)\ [N]}]^g = g(f)(g(x_i))$ iff $\forall y[g(f(y)) \in [[N]]^g$ and undefined otherwise

However, this still does not give us the right meaning in the case that the head contains a pronoun which we try to "bind" to a quantified NP in another relative clause. Consider the interpretation of *the relative of his that every man invited.* The reconstructionist's hope is that *his* can be an ordinary variable bound in the reconstruction structure by *every man*: the structure is thus:

(18) the *relative of his* [every man_i invited $t_{f(j)\ [\text{relative of his}(i)]}$]

The interpretation of the trace on an assignment function g is $g(f)(g(x_j)$ provided that the range of the relevant function is the set of relatives of $g(x_i)$. In this case x_i is ultimately bound by *every man.* Thus the entire NP has the following meaning: it denotes the unique (contextually salient) function f such that every man is an x who invited f(x) and the range of f is relatives of x. This means that for all y, f maps y into a relative of x (not, as one would hope, into a relative of y). The only situation in which any function could satisfy that is one where everyone was related to everyone else - clearly not a requirement for truth in (6). Incidentally, even if we could find some other way to get the functional meaning to come out right, a reconstruction analysis has no obvious way to block the incorrect individual meaning discussed above.

4.2. The questionable logic of the "c-command" restriction

We should, moreover, ask *why* anyone might think about a reconstruction analysis for binding in the first place. (Notice that if one is happy with a theory which does not respect direct compsositionality, then one could instead try to raise the binder at LF to scope over the pronoun. The obvious ways to do this actually give the wrong meaning - but, as shown above, so does reconstruction.) The desire to reconstruct here seems to stem from an assumption about the mapping between syntax and semantics: in order for a "binder" to "bind" a pronoun the former must at some relevant level c-command the latter. This in turn is motivated by Weak Crossover (herearter, WCO) phenomena. In other words, any theory must account for the empirical fact that (19) is bad while (6) and (8) are not:

(19) *His_i mother loves every man_i.

Thus a standard view of WCO assumes a constraint to the effect that a "binder" must c-command a "bindee" at some relevant level of representation (cf., Reinhart, 1983). Of course we need here a definition of "binder" and "bindee": assume that a binder/bindee pair is any pair of full NP (or, DP) and pronoun which are co-indexed. Assume further that the role of the indices is that there is some level of

representation at which the binder is out of the main sentence (e.g., a level derived by QR), but where its surface position is instead occupied by a pronoun or trace with the index that the binder bears at surface structure. The semantic work of co-indexation is done by the assumption that all pronouns with index i and all traces with index i translate as the variable x_i.

Consider now the following question: what is the level of representation at which the "binder" must c-command the "bindee"? Obviously this cannot be a requirement on LF - the fact that the "binder" must c-command the "bindee" at LF is true by definition (given standard assumptions about how LF trees input the semantics, there is no coherent way to give a semantics of "binding" without this.) Because of this fact, an LF condition is of no use in accounting for a typical WCO case like (19) for it can trivially be assigned an LF in which *every man* is "raised" and c-commands the pronoun. The usual assumption, then, is that the c-command requirement holds for surface structure.

But of course now we must ask whether (6) and (8) satisfy the WCO requirement under a reconstruction solution. Here the discussion depends on exactly what version of reconstruction we adopt. For starters, take the standard GB approach whereby the reconstruction level results from moving material back into the position which it occupied before it moved in the syntax. Then obviously WCO - if taken to be a constraint on *surface structure* - is still violated in (6) and (8). After all, on the *surface* the head is not in its pre-movement position and so on the surface the binder is not c-commanded by the bindee. The c-command requirement is met only for the "reconstruction level" - but what is that level? We can say it is LF but we have just seen that WCO can't be a constraint on LF. We would thus have to conclude that WCO is a constraint on some other level - call it level A - a level which is neither surface structure nor LF. Material is put back into the gap position at level A for the purpose of checking WCO violation. But since A is not LF it would appear to do no other work and have no other motivation.

The situation is slightly improved under the copy theory of movement but there still remain open questions. The idea here is that moved material remains in its pre-movement position in surface structure - but is unpronounced there. The "moved" material is actually a copy of the other material. Hence the pronoun which remains in the pre-movement position and is unpronounced is, indeed, c-commanded by its binder at surface structure. But wait - there is another pronoun (the one in the "raised" copy) which violates the WCO requirement. Surely we can solve this by defining "bindees" as only the unpronounced ones, but why should this be? (I am certain that a defender of the copy theory of movement is thinking here that the reason is obvious: the semantics will be set up in such a way that the one which actually feeds the interpretation is the unpronounced one. But this merely raises two other questions. First, why is this so? Why does the semantics interpret the original and not the copy? Second, we have seen that LF - i.e., the level which is interpreted - is not relevant to WCO. WCO is not a requirement about semantic composition but about the syntactic configuration in which we can have co-indexation. In view of this, the fact that one pronoun is "visible" to the semantics and one isn't is completely irrelevant to the syntax.) The bottom line is that reconstruction provides no insight into why (6) and (8) are good while (19) is not.

4.3. A new empirical problem for "reconstruction"

Finally, there is a striking set of facts which present new problems for reconstruction. To develop this, note first that the binding works just as well if the two clauses are reversed (I thank Chris Barker for this observation):

(20) a. The (only) woman that he loves that no man/every man would (fail to) invite is his mother.
b. The (only) woman that no man/every man would (fail to) invite that he loves is his mother.

The reconstruction solution discussed above accounts for (a) by positing that a level of representation in which *that he loves* is in the position of gap following *invite.* The result of this is that *he* is c-commanded by *no man* (or,*every man)* and can hence be unproblematically bound.. But what about (b)? Notice that - assuming a stacking structure for (10b) on the surface, *no man* does not c-command *he.* But it does not do so at the reconstruction level either: here *woman that no man loves* would be in the gap position, and would still fail to c-command *he.*

But the plot thickens: there may be a solution available to the reconstruction approach. This derives from noticing that any time we have two relative clauses - such as a simple case like (21) - there are conceivably two different structures:

(21) every man who Bill likes who Mary invited

The first is the stacking structure which we have been dealing with throughout. But - at least given a certain set of assumptions - it may be that (21) also has an extraposition structure - whereby *who Mary invited* is extraposed modifies the first relative pronoun *who* and is extraposed from this (either after *who* is fronted, or before - in which case it is extraposed from the position of the gap). Whether or not this is possible depends on a variety of other assumptions, but it is certainly not out of the question to think that (21) does have one such analysis. (For detailed discussion, see Jacobson (1982) who in fact proposes that the extraposition analysis is the *only* available analysis for (21) and that there is no such thing as stacking. If this is correct then the basic point in this section will continue to hold.) This provides a simple way to account for the binding of the pronoun by *no man* in (20b). Assume that *that he loves* is extraposed - and hence was originally part of the material that occurred in the position of the object of *invite.* Assume further that extraposed material - like any other moved material - is its original position at the reconstruction level. Then the pronoun is c-commanded by its binder at the reconstruction level. In other words, since (21) conceivably has both a stacking and an extraposition structure, we can account for both binding patterns. (20a) is an instance of stacking, while (20b) is an instance of extraposition.

Yet there is a rub. By way of background, Engdahl (1986) notices that - with respect to the analogous case of functional questions - there can be any number of pronouns with any number of binders. The same os true for relative clause cases. Thus consider a context in which all of the phonology professors are women, all of the students are men, and each student takes a phonology course from each professor. In this context, we can (22) with two binders and two pronouns:

(22) The assignment that every student most wanted every phonology professor to like was the last one he handed in to her.

Here we simply have two binders in the pre-copular constituent and two pronouns in the post-copular constituent. But we can expand on this example to create the situation of binding into a head. Such a case would be (23). Similarly, we can have the reverse binding pattern, where the relative clause containing the pronouns follows the one with the "binders" as in (24):

(23) The assignment that he gave her that every phonology professor most praised every student for was the last one he handed in to her.

(24) The assignment that every student gave every phonology professor that she most praised him for was the last one he handed in to her.

So far these cases tell us nothing new. The reconstruction analysis need only posit that (23) is an instance of (hence both pronouns are c-commanded by the binders at the reconstruction - assuming, as is independently needed - that a direct object c-commands into a PP), while (24) is an instance of extraposition.

Note, though, that we can also mappily mix and match the binding patterns:

(25) The assignment that every student gave her that every phonology professor most praised him for was the last one he handed in to her.

To account for the binding of *her* using a reconstruction solution, we need to assume that this is a case of stacking so that *assignment that every student gave her* is in the position of the gap following *for*. This way, *her* is c-commanded by *every phonology professor* at the relevant level. But this leaves *him* unbound. In order to successfully bind *him,* we need to assume that instead this is an extraposition case - whereby the second relative clause is extraposed from the gap position in the first relative clause (and hence *him* is c-commanded by *every student).* But then *her* is not in the right place to be bound. In other words, there is no obvious way to construct some representation in which both pronouns in (25) are c-commanded by their would-be binders.

5. The proposed solution under Direct Compositionality and Variable-Free

We now return to the variable-free account of (6) and (8). Recall that in the case of (3) *(the woman who every man loves is his mother)* the functional reading of the pre-copular NP was derived by intersecting the set of functions with range woman' with the set of functions that every man z-loves. Our problem stems from the expanded examples in (6) and (8). In (6), the head *relative of his* does not denote a set of functions of type <<e,e.>,t> - and it does not meet the input for the rule in (4) either. In the variable-free view it denotes a two-place relation. This follows because any expression containing an unbound pronoun is a function from individuals to the type that that expression would have were the pronoun replaced by a full NP. In other words, since *relative of Bill's* is of type <e,t>, this means that *relative of his* is of type <e,<e,t>> (and has the same meaning as the basic relational noun *relative of*). A similar problem arises in the case of (8). Here we need to worry about how to fold in the semantic composition of the extra relative clause *who he loves.* Given that a normal relative clause is of type <e,t>, a relative clause like this one which contains a pronoun is of type <e,<e,t>>.

Before continuing, a brief comment is in order concerning the actual meaning of *who he loves* under the variable-free view. The particular 2-place relation that this denotes depends on just what fine-grained assumptions one makes about how the meaning of relative clauses are put together. Under one view, the meaning for this turns out to be $\underline{\lambda x[\lambda y[\text{loves}'(x)(y)]]}$ or, more simply, loves'. This is essentially the result that one gets using a Steedman-style approach to extraction combined with the conventions for pronouns developed in Jacobson (1999; see that paper for discussion of this point). To clarify, let A|B be any expression A with a B-type gap. (Of course in Steedman (1987) there is no distinguished kind of "slash' for an extraction gap, and this would simply be an expression of category $A/_R B$. I will, however, continue to use A|B as a notation for something with an extraction gap of category B so as to be able to phrase the discussion in more general terms.) Now if the conventions for extraction and for

the passing of the pronoun-containing information are such that the final category ends up being $S^{NP}|NP$ then the meaning shown above will be the meaning for *who he loves*. One can convince oneself of this by noting that under this view the extraction gap is the expected first argument of the function and the slot occupied by the pronoun is the expected second argument. Thus the argument structure for gap in object - pronoun in subject is exactly the same as for the ordinary transitive verb *love*. There are, however, other accounts of extraction with in CG (see, e.g., Jacobson 1989, Oehrle 1991), which are such that interacting these with the pronoun conventions of Jacobson (1999) yields a final category $(S|NP)^{NP}$. Here the argument slot occupied by the pronoun (the subject slot) is the expected first argument in while the extraction gap here (and in all other cases) is *the innermost argument slot*. Under this view, then, the meaning of *who he loves* will be the reverse - i.e., $\underline{\lambda x[\lambda y[loves'(y)(x)]]}$. As will be documented below, this is exactly what we need the meaning to be (the reason will emerge below) - thus we would hope that the second class of views on extraction are the right one rather than exactly a Steedman-style view. But this turns out to be an extremely fortuitous result. The interaction of extraction and pronouns is discussed in detail in Jacobson (1999, fn.19) and - for totally independent reasons - I came to the same conclusion there.

Returning to the main theme, we are trying to come up with a story regarding the semantic composition of *the woman who he loves that every man invited*. We have seen that *who he loves* denotes a 2-place relation, while *woman* and *that every man invited* can both denote sets of functions, where these two sets will intersect. How can we fold a 2-place relation into the semantic composition? It appears that we need some new type-shift (or, combinatory) apparatus to do this.

The solution that I wish to propose does indeed involve a new type-shift rule above and beyond the other mechanisms proposed in Jacobson (1999, 2000) (which used only the **z** rule and the **g** rule). But, it will turn out that this type shift rule is both simple and very natural. We begin with the type of case in (6) (*the relative of his that no man invited).* All will be straightforward if there was some way for *relative of his* to denote the set of functions $\underline{\lambda f[\forall x[relative\text{-}of'(x)(f(x))]]}$ (this is the set of functions which map each individual into a relative of theirs). This meaning folds into the semantic composition in the obvious way (because - like the other bits - it is a set of funcitons), and it gives us the right truth conditions. As we have seen, *relative of his* is "born" denoting the 2-place relation mapping any x and y to true iff x is a relative of y. Now consider what happens if we try to shift this into a set of functions in the most simplistic and minimal way: the above is exactly what we get. To see this, take any function F of type <e,<e,t>>. There is a natural mapping from F to a set of functions; this can be shown by first de-Currying F into a set of order pairs of individuals and then collecting up all the subsets of this set which are functions. (It should be noted that there are actually two ways we could de-Curry F depending on which we chose to be the first member of the ordered pair and which the second. The correct result requires that we de-Curry in such a way that outermost argument slot of F is the first member of the ordered pair while the next argument slot is the second member, though I know of no reason why it should have turned out this way over the other possibility). Once this proceedure is carried out for the case of *relative of his*, we have exactly the set of functions shown above.

Formally , then, assume that we can define an operation **m** as follows:

(26) Let F be a function of type <b,<a,t>>. Then **m**(F) is a function of type <<b,a>,t> such that $\mathbf{m}(F) = \lambda h_{<b,a>}[\forall x_{\text{in the domain of h}} [F(x)(h(x))]]$, where h is a partial function from b to a.

Note that (26) may look arbitrary, but it is inded the "natural" mapping from 2-place relations to sets of functions defined above. Note further that, as mentioned above we could as well have de-Curry'ed in the reverse way, which is to say that the **m** operation could just as naturally have been stated as in (27):

(27) Let F be a function of type <b,<a,t>>. Then **m**(F) is a function of type <<b,a>,t> such that **n**(F) = $\lambda h_{<a,b>}[\forall x_b[F(h(x))(x)]]$

The evidence that (26) is the correct version and not (27) comes primarily from the pre-shifted meaning of *relative of his*. We know that this denotes the 2-place relation $\underline{\lambda x[\lambda y[\text{relative-of'}(x)(y)]]}$ which is why would need to choose (26) rather than (27It is this fact in turn which leads to the conclusion discussed above that *who he loves* must have the meaning $\underline{\lambda x[\lambda y[\text{loves'}(y)(x)]]}$ and not $\underline{\text{loves'}}$, which in turn means that its syntactic category is $(S|NP)^{NP}$ and not $S^{NP}|NP$. As noted above, this is a very happy result in view of the independent evidence for this discussed in Jacobson (1999). To complete the account, this means that the grammar contains the rule in (28); note that I am oversimplifying in that I ignore the syntax connected to this rule:

(28) Let α be an expression of the form $<[\alpha]; \ldots ; \alpha'>$, for α' a function of type <a,<b,t>>. Then there is an expression β of the form $<[\alpha]; \ldots ; \mathbf{m}(\alpha')>$ (syntax is ignored here)

This apparatus allows *relative of his* to denote the set of functions $\underline{\lambda f[\forall x[\text{relative-o f'}(x)(f(x)]]}$. Moreover, a relative clause like *who he loves* will shift from $\underline{\lambda x[\lambda y[\text{loves'}(y)(x)]]}$ to the set of functions $\underline{\lambda f[\forall x[\text{loves'}(f(x))(x)]]}$. There is, therefore, now no problem with giving the semantic composition for an NP like *the woman who he loves that every man invites. woman* denotes the set of functions whose range is the woman set; *who he loves* denotes the set of functions mapping each member of its domain into someone who they love, and *that eveyr man inv*ites is the set of functions f such that every man z-invites f. These three sets can happily intersect. Notice too that exactly the same thing will hold if the relative clauses are reversed as in *the woman that every man invited that he loves.* Whether or not there is an extraposition structure for this is irrelevant - provided that stacking is possible then one possible analysis here is the stacking one. And there is no problem in giving a meaning for this under the stacking analysis - again the three sets intersect. The order in which we do the stacking makes no difference. The moral, then, is that there is no actual "binding" relationship between *every man* and *he.* As in other cases in variable-free semantics, "binding" is an illusion - the semantics just smoothly works to give the appropriate meaning.

This solution turns out to have a striking affinity to an analysis developed independently and proposed for a related but different range of facts in Winter (this volume). In fact, the two analyses are inverses: Winter proposes a type-shift rule mapping a set of functions into a 2-place relation. There are interesting differences between the two. As Winter points out, the mapping from a 2-place relation to a set of functions preserves information (we can uniquely recover the input from the output), while this is not the case for the reverse mapping. (Many different sets of functions will "flatten out" to the same 2-place relation.) This observation is used by Winter to predict certain facts about the set of possible determiners which can occur with functional NPs. Should Winter's version be the correct one I believe that the analysis given above regarding the binding facts will continue to go through with just the predictable adjustments (we put the meanings

together not by intersecting sets of functions but by doing a generalized intersection on the relations). I will, however, not explore this here

6. Interaction with Weak Crossover

Weak Crossover effects show up in one of the relative clauses but not the other:

(29) a. The (only) woman who he loves that no man invited was his mother.
b. *The (only) woman who he loves that invited no man was his mother.

(30) a. The (only) woman who he loves that no man invited was his mother.
b. The (only) woman who loves him that no man invited was his mother.

This follows straightforwardly under the present analysis. Consider first the contrast in (29). As noted in Jacobson (1994), this is simply analogous to the parallel case for functional questions, whose connection to WCO was first pointed out in Engdahl (1988):

(31) a. Who does every man love? His mother.
b. *Who loves every man? His mother.

Engdahl (1988) observed that these reduce to WCO effects given her analysis of functional questions (see also Chierchia, 1991). Of course the precise details depend on the particular mechanics used for functional gaps in general, but Engdahl's idea was roughly that in a functional question there is a gap translating as $\underline{f(x)}$. In (31a) the variable $\underline{x}$ can be bound by the subject *every man* in the normal way that binding takes place. In (31b) the variable cannot be bound by the object for whatever precise reason accounts for WCO effects in general. Of course under variable-free semantics the story needs to be translated into a different language, but the basic observation goes through without a hitch. As discussed in much greater detail in Jacobson (1994, 1999), the explanation for run-of-the-mill WCO effects is that the "binding" rule **z** is formulated so as to only "merge" a pronoun within one argument to a later (or, "higher") argument slot. Hence, for example, a pronoun within an object can be "merged" with the subject slot. But there is no backwards rule **s** whose effect would be to merge a pronoun within one argument to a lower (or, earlier) argument slot. (Hence there is no way to merge a pronoun in subject position with the object argument slot.) Since functional readings are simply the result of application of **z** on the verb, the contrast between (29a, 31a) and (290b, 31b) is expected. In (29a, 31a) *invite* has undergone **z**, with the effect that the relative clause is the set of functions that every man z-invites. (29b, 31b) does not allow for an analogous functional reading as this would require **s** on *invite* (where the relative clause would then denote the set of functions f such that f s-invited every man).

But in the cases in (30) we find no WCO asymmetry. This is because here the relationship between *he* and the gap is not given by the **z** rule but rather by **m**, and **m** has no difficulty operating in either of these cases.

(32) a. (=31a) *who he loves;* $(S/_R NP)^{NP}$; $\lambda x[\lambda y[loves'(y)(x)]]$ $--->_m$ $\lambda f[\forall z[loves'(f(z))(z)]]$
b. (=31b) *who loves him;* $(S/_L NP)^{NP}$; loves' $--->_m$ $\lambda f[\forall z[loves'(z)(f(z))]]$

Apropos WCO, we saw earlier that it was not trivial under a reconstruction approach to account for the contrast between a standard WCO violation like (11)

and good cases like (6) and (8). In neither case does the binder c-command the (pronounced) bindee at surface structure, and in both cases it does so at LF, so we need some way to distinguish these. But under the analysis here, no new twists are needed. A standard WCO violation like (11) is bad because it would involve illegitimate use of a type-shift rule **s**. (6) and (8) are fine because they involves **m**. The apparent "binding effect" results from two different type-shift processes. (Granted, we need to posit the second "binding" type-shift rule **m**, but we are not invoking extra machinery just to get WCO to come out right.)

7. Comparison to Engdahl

As it turns out, the solution proposed here has striking similarities to the proposal (for the analogous case of functional questions) developed in Engdahl (1986). This may not obvious at first glance as her implementation had wrapped into it certain facts which are particular to questions. But if we tease out certain irrelevant differences between Engdahl's account and the present one, we can recast the essence of her proposal by positing a type-shift rule as in (33), although the precise interpretation of this rule is what we will be addressing momentarily:

(33) P ---> $\lambda f[\forall x[P(f(x))]]$ (for P of type <e,t>)

Crucially, $\underline{x}$ here is a variable over variable names (not over individuals). The variable names play a crucial part in the system, and (33) is a rule schema allowing any variable name to be chosen. Indeed, the rule is deliberately set up in such a way that it can "capture" an unbound variable within the input - and it is this which makes this interestingly different from the variable-free analogue. In the latter, a phrase like *relative of his* denotes a two-place relation between individuals and shifts by **m** to denote a set of functions. In a system with variables, *relative of his* is not a function of type <e,<e,t>>. Rather it is - relative to an assignment function - a function of type <e,t>. Let us suppose its meaning is what we will represent as $\underline{\text{relative-of}'(x_i)}$. Then if $\underline{x_i}$ is the variable chosen in an applicaton of P, this maps into $\underline{\lambda f[\forall x_i[P(f(x_i)]]}$. Relative to some assignment function, this is the set of functions f such that everyone is a relative of f(x). Since all the variables are bound here this will actually have the same value on all assignment functions

At first glance, this appears to have an advantage over the mechanisms which we have set up in the variable-free system. For (33) kills two birds with one stone. Note that this looks much like the rule given in (4) in the variable-free system - (4) is the rule allowing the meaning of a head noun like *woman* to shift in such a way that it denotes a set of functions whose range is $\underline{\text{woman}'}$. In the variable-free system we need both this rule and the **m** shift rule. I see no way to collapse them because their inputs are different - (4) operates on functions of type <e,t> while **m** operates on functions of type <e,<e,t>>. But in a system with variables both *woman* and *relative of his* are functions from assignment functions to functions of type <e,t> and so a single rule will effect the requisite shift in both cases.

Nonetheless, this collapsing is at a heavy cost. This becomes clear only once we consider just what is the actual status of (33). Engdahl was also working in a theory which assumes direct compositionality but where variable names play a crucial role. Thus her analogous rule was also intended as a rule mapping model-theoretic objects to model-theoretic objects. (Recall again that (33) was not exactly the rule that Engdahl proposed, but is in the spirit of her proposal.) But then, just what is this mapping? In the variable free framework, the analogous rule (4) mapped a set of individuals into a set of functions, while the additional **m** rule mapped a 2-place relation to a set of functions. (33) does neither. What it does is

actually map a functions from assignment functions to functions of type <e,t> to a function from assignment functions to sets of functions.

Now in the case where it applies to the meaning of *woman* (whose meaning is a function from assignment functions to <e,t> functions), the use of the assignment functions is really irrelevant. *woman* contains no unbound pronoun within it, and so its meaning is a constant function from assignment functions - it maps all assignment functions into the set of women. After the shift, we also have a constant function - each assignment function is mapped into the set of functions whose range is the set of women. So far, this is just like the variable-free analogue except with the extra step of having each model-theoretic object be a function from assignment functions. But since we are dealing with constant functions in this case, we can "forget about" the assignment functions - and once we do that the rule is just as natural as is the rule in (4) (and, in fact, it is the same rule).

But crucially we do not have the luxury of "forgetting about" the assignment functions in the case that the rule maps $\underline{\text{relative-of'}(x_i)}$ to $\underline{\lambda f[\forall x_i[\text{relative-of'}(x_i)(f(x_i)]]}$. There is no way to think about the model-theoretic status of this shift without taking into account the assignment functions. This is because this rule works in this case only because it captures a variable which is unbound in the input. Thus to see the actual model-theoretic content of (33), take that subset of the set of assignment functions which agrees on assignments to all variables except some variable $\underline{x_i}$. Call that subset H. Then the input to (33) is a (possibly non-constant) function from H to sets of individuals, while the output is a constant function from H to sets of functions. For any assignment function g in H, the input is some set of individuals, and the output is the set of functions f such that for all assignment functions g' in H, the set of individuals that the input expression assigns to g' includes the set of individuals that f assigns to the value of $\underline{x_i}$ on g'. I see no way to see this as any kind of "natural" shift. It lacks both the simplicity and the naturalness of a rule like **m**. I think it is fair to say that Engdahl's rule has always had the feeling of involving a "trick" - it does the job, it can be stated as a direct mapping from model-theoretic objects to model-theoretic objects, but it seems entirely arbitrary. I would like to propose that this arbitrariness stems solely from the mistake of embedding it in a view with variables. Once Engdahl's solution is translated into the variable-free framework as is done here, the naturalness of this shift becomes quite apparent.

8. Generalizing to the Engdahl n-place cases

The apparent death-knell for the reconstruction solution was (25) - the case with a mix-and-match binding pattern. The variable-free solution proposed above cannot, therefore, claim total victory until it is shown that it can account for this. First let us deal with simpler cases where there are two (or more) pronouns within one of the relative clauses, and two or more "binders' within the other, as in (23):

(23) The assignment that he gave her that every phonology professor most praised every student for was the last one he handed in to her.

We have three tasks here: (i) the head must shift into a set of functions from two individuals to individuals (i.e., to an object of type <<<e,<e,e>>,t> ; (ii) the relative clause *every professor most praised every student for* must be able to compose in such a way as to yield an object whose meaning is of this type; and (iii) similarly for the relative clause *that he gave her*.

Beginning with (ii), this is shown already in Jacobson (1999). The basic idea is simply to let *praise (for)* undergo two successive applications of **z** (one is

the version of **z** which allows the subject argument slot to "bind" while one allows the direct object argument slot to "bind"). For simplicity I treat *praise for* as a single verb (this is not necessary except for expository purposes). Thus the lexical meaning of this is such that it expects an individual argument in the position following *for*; it is thus of type <e,<e,<e,t>>> and the "gap" following *for* would - in a non-functional relative of this kind - just be an individual gap. But the verb can shift by **z** so that the position following *for* is expected to be a function of type <e,e>. One further shift will turn this position (i.e., the "gap") into a function of type <e,<e,e>>. The final verb, then, is of type <<<e,<e,e>>,<e,<e,t>>> . In other words, it wants as object of *for* a function from two individuals to another individuals (in this case, the final individual is an assignment). Then it wants a direct object which is ultimately occupied by *every student* and finally it wants a subject which in this case is *every phonology professor*. But in (23) it doesn't get its first argument - the general rules for forming relative clauses are such that it can hold off on this argument position, and the final result here is a relative clause of type <<e,<e,e>>,t>.

There is one further point to note here. In order to allow the gap to have the complex type, we perform two applications of **z**. This causes the gap to be a function from two individuals (ultimately in this case a student and a professor) to an individual (in this case, ultimately an assignment). One of the individual argument slots of this complex function is "merged" to the subject slot of *praise for* (the slot occupied by *every professor*) and one is "merged" to the direct object slot of *praise for* (the slot occupied by *every student*). This is allowed because **z** is formulated in such a way that when it operates on a 3-place verb, either binding possibility is allowed. (See Jacobson, 1999 for the full generalized form of **z**.) Moreover. the two different versions of this operation can happen in either order - and this will in turn give the result that there are actually two different meanings which can be associated with this relative clause. To clarify, let me notate the type of the "gap" as $<e_1<e_2,e>>$. In one meaning, the subject slot (*every student)* binds the e_1 position while the object slot (*every professor)* binds the e_2 position, the other meaning has the reversed binding pattern. (Actually, there are two other possible meanings with this same type - both argument positions can be bound by the subject slot or both by the object slot. This is exactly what we need for cases where, for example, there is an actual object which contains two pronouns bound by the same thing. But cases of this kind do not concern us here.)

The next task is (i) above - to show that the head can shift to denote a set of functions from two individuals to an individual (in this case, the relevant function when all is put together ends up being a function from professors and students to assignments). But this simply follows from recursive application of the type shift rule given in (4):

(34) assignment_1 of type <e,t> --->
assignment_2 of type <ee,t>: $\lambda f[\forall x[\text{assignment}_1'(f(x))]]$ --->
assignment_3 of type <<e,ee>,t>: $\lambda F_{<e,ee>}[\forall y[\forall x[\text{assignment}_1'(F(y)(x))]]]$

Thus tasks (i) and (ii) required no new apparatus in the system. In order to complete task (iii) (assigning the double functional reading to *that he gave to her)* we need to generalize the rule **m**, but in a way which is perfectly obvious and unsurprising. We simply need to reformulate **m** so that it can apply recursively:

(35) a. Let F be a function of type <b,<a,t>>. Then **m**(F) is a function of type <<b,a>,t> such that **m**(F) = $\lambda h_{<b,a>}[\forall x_{\text{ in the domain of h}}[F(x)(h(x))]]$

b. Let F be a function of type <c,<b,<a,t>>>. Then **m**(F) is a function of type <c, <<b,a>,t>> such that **m**(F) = λC_c[**m**(F(C))]

There is nothing unexpected about this generalization. First note that it is exactly analogous to the recursive definition of **g** which is needed independently (see Jacobson, 1999, p. 138 for the recursive formulation and discussion of **g**.) A generalization of this type simply allows us to "hold off" outermost argument slots and perform operations on the inerior portions of complex functions. Moreover, Barker (personal communication) notes that this is actually nothing more than "Geaching" the operations (thus (x35xb) is actually **g(m)**.)

Given this generalization, we can obtain the appropriate meaning for *that he gave her*. To show this, I will assume that the argument structure for *give* is such that it first combines with the second object (which in this case is a "gap" which corresponds to the head noun *assignment)*, then with the direct object (*her)* and finally with the subject. In other words, it is assumed here that in a full version such as *The student gave the professor an assignment,* I assume that *an assignment* is introduced first, and that *the professor* then 'wraps" in (see Bach, 1979, 1980; Dowty, 1982. This, however, is not crucial - everything will work out just as well if the post-verbal arguments are introduced in the opposite order. Moreover, the conventions for pronouns discussed in Jacobson (1999) are such that in something like *he gave her the book,* the type is <e,<e,t>> where the subject position (occupied by *he)* is the outermost argument position. (This is confusing and is necessary to clarify for the exposition, but again it really doesn't play a crucial role in the point here.) All that is crucial is the assumption which was discussed in Sec. 5- the conventions for forming relative clauses are such that the gap slot *(that) he gave her* __ ends up as the innermost argument position of the entire relative clause. Thus we begin with a relative clause whose meaning is a 3-place relation among individuals (i.e., it is of type <e,<e,<e,t.>>>) and in the first step it undergoes a shift by **m** as shown in (36):

(36) *that he gave her:* $((S/_R NP)^{NP})^{NP}$; $\lambda x[\lambda y[\lambda z[gave(z)(y)(x)]]]$

$\rightarrow_{\textbf{generalized m}}$ $\lambda s[\mathbf{m}(\lambda x[\lambda y[\lambda z[gave(z)(y)(x)]]](s))] =$

$\lambda s[\mathbf{m}(\lambda y[\lambda z[gave(z)(y)(s)]])] =$

$\lambda s\ [\ \lambda f[\forall x[\lambda y[\lambda z[gave(z)(y)(s)]](x)(f(x))]]] =$

$\lambda s[\ \lambda f\ [\forall x[\ gave\ (f(x))(x)(s)]]]$

We can clarify this as follows. The relative clause composes up to be of type <e,<e,<e,t>>>. The gap is the expected last argument in, and so it is the one which ultimately corresponds to an assignment. We can thus illustrate the pre-shifted meaning of the relative clause as $<e_{students}, <e_{professors}, <e_{assignments}, t>>>$. Of course the grammar does not restrict the argument slots in the way shown here; I am putting in these subscripts just to aid in keeping track of things. The first step "holds off" on the subject slot (the student-slot) and performs **m** on the interior, mapping the above to a function of type $<e_{students}, <<e_{profs}, e_{assignments}>,t>>$. We will now perform a second application of **m** in the straightforward way. Our input meaning here is a 2-place relation between individuals and functions (of type <e,e>), and so we collect this up to a set of functions from individuals to functions (of type <e,e>). In other words, our final product will be of $<<e_{students}, <e_{profs}, e_{assignments}>>,t>$ (where again the subcripts are just to help us keep track of things); the full details are:

(37) $\lambda G_{<e,ee>}[\forall y[above(y)(G(y))]] =$

$\lambda G[\forall y[\lambda s[\ \lambda f\ [\forall x[\ gave\ (f(x))(x)(s)]]](y)(G(y))]] =$

$\lambda G[\forall y[\lambda f\ [\forall x[\ gave\ (f(x))(x)(y)]]\ (G(y))]] =$
$\lambda G[\forall y[\forall x[\ gave\ (G(y)(x))(x)(y)])\ ]]$

To complete the analysis of (23) and (24) the head and both relative clauses all denote sets of functions of type <e,<e,e>> and so can happily intersect.

But our real mission is to show that the analysis here can also handle the mix and match case in (25), which was especially problematic for reconstruction.. Thus we now have a new task: to demonstrate that the apparatus applies to a relative clause like *that every student gave her* so as to also map this into a function of type <<e,<e,e>,t>>. To give the broad outline of the discussion before proceeding: this in itself will turn out to be straightforward. But a subtle problem emerges when we put together the full meaning of a mix and match case. This problem will require us to adopt a second generalization of **m**. Thus - in all fairness - the analysis here does not *automatically* get the mix and match case. On the other hand, we need only generalize **m** slightly to get this- thus the general program has no real difficulty here. A reconstruction solution, on the other hand, simply appears to have no way to account for such a case (at least not without tremendous complications) - the existence of the mix and match pattern is simply antithetical to the idea that we can solve the binding problem by putting things into the right place at the right level.

Thus consider the composition of the meaning of *that every student gave her*. This needs no new apparatus. Here *gave* undergoes **z** so that expected argument (the gap) is of type <e,e> . Note that this is the version of **z** which allows the subject argument slot to be merged with the argument slot of the <e,e> function. This is just what happens in the case a simple functional relative clause. Take, for example, the underlined portion of *the assignment <u>that every student gave Professor Jones</u> (was the one he had worked on all night).* This relative clause can be of type <<e,e>,t> - i.e., it can denote a set of functions. Now in the case at hand, the direct object slot is instead a pronoun and so the ultimate type of the relative clause has one more argument slot - and so it is of type <e,<<e,e>,t>>, (Once again it is worth recalling that the argument structure of a relative clause is such that the gap position is the innermost argument slot - hence the type shown above.) We need only one final step: the meaning of the relative clause shifts by **m** to give us a meaning of type <<e,<e,e>>,t>. This, incidentally, is no different from what we saw in the basic kind of case that we have been dealing with throughout. Take a relative clause like *who he loves.* This shifts from <e,<e,t>> to <<e,e>,t>. Let me use f as an abbreviation for <e,e> . Then here we begin with something of type <e,<f,t>> and shift it to a meaning of type <<e,f>,t>. The full details, are shown in (38):

(38) *that every student gave her* __: $(S/_R NP^{NP})^{NP}$; $\lambda x[\lambda f_{<e,e>}[$every-student'(**z**-gave(f)(x))]] = $\lambda x[\lambda f_{<e,e>}[$every-student'$(\lambda y[$gave(f(y))(x)(y)])]]
$\rightarrow_{\mathbf{m}}$ $\lambda G_{<e,ee>}[\forall z[$ above (z)(G(z))]] =
$\lambda G_{<e,ee>}[\forall z[$every-student'$(\lambda y[$gave(G(z)(y))(z)(y)])]]

Now let us consider the composition of the second relative clause *that every professor most praised him for.* One might think we are home free - we compose this in a fashion exactly analogous to that shown in (39). Indeed, this is a legitimate derivation - but it is not the one we need in order go give the pragmatically felicitous meaning in (25). Actually the discussion here is complicated by the fact that gender contributes something to the meaning - and so, pragmatics aside, the possible "binding" patterns in (25) are constrained by the gender of the pronouns. In order to ignore this complication, we recast this as (39) - in a context where students and professors are all women:

(39) The assignment that every student gave her that every phonology professor most praised her for was the last one she handed in to her.

The meaning of the first relative clause - put together by **z** on give and later **m** on the entire relative clause gives the result shown in (38). If we put the meaning of the second relative clause together in an analogous fashion, its meaning is:

(40) $\lambda G_{<e,ee>}[\forall z[\text{every-professor}'(\lambda y[\text{praise-for}'(G(z)(y))(z)(y)])]]$

The meanning of the post-copular constituent is $\lambda p[\lambda s[$the last assignment that s handed in to p$]$. The full semantics equates says that this function is in the set of G functions described in both (38) and (40). For (38) this is a good result: the givers of the assignment are also the ones who hand it in and the givees are the handees-in. But in (40) things are reversed, and so the handers-in of the assignment will be the praisers.

This is reminiscent of a case discussed in detail in Jacobson (1999): in the mix-and-match case here one of the relative clauses wants their "binders" to "bind" in different orders. To solve this, we need to be able to have two different orders of application of **m** and **z**. As such, we need to first compose VP *praise him for* to give a function of type of type $<e_3,<e_2,<e_1,t>>>$. I notate the argument slots for convenience: e_1 (the outermost slot) is, as usual the "gap" slot and e_e is the slot occupied by the pronoun, while e_3 is ultimately the subject slot. We then want to be able to shift by this meaning by a generalization of **m** to give a function of type $<<e_3,e_2>, <e_1,t>>$. (Thus this is looking for the kind of thing that could be a function from, for example, students to assignments, and it gives back an $<e,t>$ function.) This in turn undergoes **z** in such a way as to introduce a new arguments slot on that first complex argument (the argument which, in this case, is ultimately a function into assignments), and it "merges" that newly created slot with the subject slot. The VP thus has the following meaning: $<<e_1, <e_3,e_2>>, <e_1,t>>$ and - when this then combines with the subject (by function composition) we have the appropriate meaning for the entire relative clause. It is a set of functions from two individuals into assignments, and it is expecting to take the arguments of those functions in the order professors - students. This is the exactly the order compatible with the meaning of the post-copular constituent. The requisite generalization is given in (41):

(41) Let R be a function of type $<e,<e,,<e,t>>>$. Then **extended-m**(R) = $\lambda f[\lambda y[\forall x[R(x)(f(x))(y)]]]$

9. Remaining Issues

There remain two pressing open questions about this analysis. First, will the addition of **m** cause a "meltdown" in the system? Will it apply too generally and predict the existence of incorrect meanings? This question cannot really be answered until the analysis here is completed with an analysis of the syntax; the hope is that the a syntactic category requirement on the input to **m** will ensure that it applies only in the appropriate places. More seriously, I have argued that **m** is "natural" - in a fairly obvious way. But what about "extended **m**? It remains to be demonstrated that this too is a natural operation.

Endnotes

*I would like to thank Chris Barker and Irene Heim for relevant discussion. I especially thank Michael Rosen for pointing out to me the "naturalness" of the relevant operation. The name **m** stems from this, but is a happy choice, it is exactly the inverse of Winter's proposal (this volume), which could well be named **w**.

References

Bach, E. (1979). "Control in Montague Grammar", *Linguistic Inquiry* ***10,*** 515-531.

Chierchia, G. (1991). "Functional WH and Weak Crossover", in D. Bates (ed.), *Proceedings of the Tenth West Coast Conference on Formal Linguistics.* Stanford: CSLI Publications.

Dowty, D. (1982). "Grammatical Relations and Montague Grammar", in P. Jacobson and G.K. Pullum (eds.), *The Nature of Syntactic Representation.* Dordrecht: D. Reidel.

Engdahl, E. (1986). *Constituent Questions.* Reidel, Dordrecht.

Engdahl, E. (1988). "Relational Interpretation" in R. Kempson (ed.), *Mental Representations: The Interface Between language and Reality.* Cambridge: Cambridge University Press, pp. 63-82.

Geach, P. (1972). *Logic Matters.* Oxford: Basil Blackwell.

Groenendijk, J. and M. Stokhof (1983). "Interrogative Quantifiers and Skolem Functions" in K. Ehlich and H. van Reimsdijk (eds.), *Connectedness in Sentence, Discourse and Text,* Tilburg Studies in Language and Literature **4**, Tilburg University, Tilburg.

Jacobson, P. (1982). *The Syntax and Semantics of Multiple Relatives in English.* Indiana: Indiana University Linguistics Club.

Jacobson, P. (1989). "A(nother) Categorial Grammar Account of Extraction", paper presented at the Conference on Categorial Grammar, LSA Summer Institute, Tucson, Arizona.

Jacobson, P. (1994). "Binding Connectivity in Copular Sentences", in M. Harvey and L. Santelmann (eds.), *Proceedings from Semantics and Linguistic Theory IV,* Cornell Univeristy, DMML Publications, pp. 161-178.

Jacobson, P. (1999). "Towards a Variable-Free Semantics", *Linguistics and Philosophy* ***22,*** 117-184.

Montague, R. (1973). "The Proper Treatment of Quantification in Ordinary English", in R. Thomason (ed.), *Formal Philosophy: Selected Papers of Richard Montague,* 247-278. Yale University Press, New Haven.

Oehrle, R. (1990). "Categorial Grameworks, Coordination, and Extraction", in A. Halpern (ed.), *Proceedings of the Ninth West Coast conference on Formal Linguistics.* Stanford: CSLI Publicaitons.

Reinhart, T. (1990). "Self Representation", paper presented at the Princeton Conference on Anaphora, Princeton.

Sharvit, Y. (1998). "Connectivity in Specificational Sentences", *Natural Language Semantics* ***7,*** pp. 299-304.

Sharvit, Y. (1999) "Functional Relative Clauses", *Linguistics and Philosophy.*

Stechow, a. von (1990). "Layered Traces", paper presented at the Conference on Logic and Language, Revfullop, Hungary.

Steedman, M. (1987). "Combinatory Grammars and Parasitic Gaps", *Natural Language and Linguistic Theory* ***5,*** 403-440.

Vergnaud, J.-R. (1974). *French Relative Clauses.* Ph.D. Dissertation, MIT.

Winter, Y. (this volume). "Functional readings and wide-scope indefinites", in B. Jackson et al. (eds.),*Proceedings from Semantics and Linguistic Theory 12.* Cornell University, DMML Publications.