

First, the simple stuff – Det-Nom, no domain restrictions:

*The woman who John loves left the party.*

[[who John loves]] =  $\lambda P[\lambda x[\text{loves}'(x)(J) \ \& \ P(x)]]$

[[woman who John loves]] =  $\lambda x[\text{loves}'(x)(J) \ \& \ \text{woman}'(x)]$

[[the woman who John loves]] =  $\iota z[\text{loves}'(z)(J) \ \& \ \text{woman}'(z)]$

[[left the party]] =  $\lambda x[\text{left-the-party}'(x)]$

[[the woman who John loves left the party]] =  $\text{left-the-party}'(\iota z[\text{loves}'(z)(J) \ \& \ \text{woman}'(z)])$

Next, the NP-S analysis, which requires folding in domain restriction:

[[who John loves]] =  $\lambda x[\text{loves}'(x)(J)]$

[[the woman]] =  $\lambda d_{\text{et}}[\iota z[\text{woman}'(z) \ \& \ d(z)]]$

*(remaining, for now, agnostic on where the domain restriction argument originates)*

[[the woman who John loves]] =  $\iota z[\text{woman}'(z) \ \& \ \text{loves}'(z)(J)]$

Let's also assume that we have some way of generating new slots for the RC or other restrictor once it's filled.

Now, functional nominals with the Det-Nom treatment:

*The woman who every man loves is his mother.*

From Jacobson 1994:

relativizers, head nouns and determiners shift to accommodate functional readings

Given normal “who”,  $\lambda P[\lambda Q[\lambda x[P(x) \ \& \ Q(x)]]]$  of type  $\langle\langle e, t \rangle, \langle\langle e, t \rangle, \langle e, t \rangle\rangle\rangle$ , there is an additional meaning for “who” of type  $\langle\langle f, t \rangle, \langle\langle f, t \rangle, \langle f, t \rangle\rangle\rangle$  (where  $f = e, e$ , a function over individuals) which is:  $\lambda G[\lambda H[\lambda f[G(f) \ \& \ H(f)]]]$

woman'  $\langle e, t \rangle$  becomes  $\langle\langle e, e \rangle, t \rangle$ :  $\lambda f[\forall x[\text{woman}'(f(x))]]$

the'  $\langle\langle e, t \rangle, e \rangle$  becomes  $\langle\langle f, t \rangle, f \rangle$ :  $\lambda G[\iota f_{e, e}[G(f)]]$

$z[\text{loves}] = \lambda f[\lambda x[\text{loves}'(f(x))(x)]]$

[[every man]] =  $\lambda P[\forall z[\text{man}'(z) \ \rightarrow \ P(z)]]$

[[every man z-loves]] =  $\lambda f[\forall z[\text{man}'(z) \ \rightarrow \ \text{loves}'(f(z))(z)]]$

[[who every man z-loves]] =  $\lambda H[\lambda g[\forall z[\text{man}'(z) \ \rightarrow \ \text{loves}'(g(z))(z)] \ \& \ H(g)]]$

[[woman who every man z-loves]] =  $\lambda g[\forall z[\text{man}'(z) \ \rightarrow \ \text{loves}'(g(z))(z)] \ \& \ \forall x[\text{woman}'(g(x))]]$

[[the woman who every man z-loves]] =  $\iota f_{e, e}[\forall z[\text{man}'(z) \ \rightarrow \ \text{loves}'(f(z))(z)] \ \& \ \forall x[\text{woman}'(f(x))]]$

so this is approximately: the function over individuals such that all men  $z$ -love  $f$  and  $f$  picks out individuals in the set of women

So now the above, but with the NP-S analysis:

*The woman who every man loves is his mother.*

Folding in domain restrictions, what is crucial here is just like we shifted the meanings for the relativizer, nominal head and determiner, the domain restriction will now be a restriction over functions over individuals,  $\langle f, t \rangle$ . So again, remaining fairly agnostic on where exactly the domain restriction argument originates, the “fancy” meaning of “the woman” would be:

$$\lambda R_{\langle ee, t \rangle} [\iota f [\forall x [\text{woman}'(f(x))] \ \& \ R(f)]]$$

“who every man loves” cannot have the meaning we gave it above (but this shouldn’t be surprising); instead, this should be similar to the meaning we gave the relative clause in the non-functional *the woman who John loves*, except of type  $\langle \langle e, e \rangle, t \rangle$ . This is simple enough if we assume that *who* is semantically vacuous and give the RC the same meaning that we gave for “every man loves”:

$$\lambda f [\forall z [\text{man}'(z) \rightarrow \text{loves}'(f(z))(z)]]$$

Now, this is just a simple matter of function application!

$$\lambda R_{\langle ee, t \rangle} [\iota f [\forall x [\text{woman}'(f(x))] \ \& \ R(f)]] (\lambda f [\forall z [\text{man}'(z) \rightarrow \text{loves}'(f(z))(z)]])$$

$$\iota f [\forall x [\text{woman}'(f(x))] \ \& \ \forall z [\text{man}'(z) \rightarrow \text{loves}'(f(z))(z)]]$$

This is exactly the same result we had with the Det-Nom treatment.

Binding into heads, Det-Nom:

(This is a review of what is covered in Jacobson 2002)

*The woman that he loves (the most) that every man invited is his mother.*

Introduction of **m**: Given  $F$ , a function of type  $\langle b, \langle a, t \rangle \rangle$ , then  $\mathbf{m}(F)$  is a function of type  $\langle a, \langle b, t \rangle \rangle$  such that  $\mathbf{m}(F) = \lambda H_{\langle b, a \rangle} [\forall x \in H [F(x)(h(x))]]$  where  $H$  is a partial function from  $b$  to  $a$ .

**m** takes a two-place relation and reduces the number of arguments the function takes, while also reversing the order of the arguments (this may seem arbitrary but see Jacobson 2002 for a brief discussion of the issue and also Jacobson 1999 for other motivations for this move). This is the only extra piece that we need to be able to account for so-called “binding into heads”.

[[that he loves]] =

$$[[\text{he loves}]] = \lambda x [\lambda y [\text{loves}'(y)(x)]]$$

$$m([\text{he loves}]) = \lambda h[\forall z[\text{loves}'(h(z))(z)]]$$

$$[[\text{that he loves}]] = \lambda G[\lambda h[\forall z[\text{loves}'(h(z))(z)] \& G(h)]]$$

$$[[\text{that every man invited}]] = \lambda H[\lambda f[\forall z[\text{man}'(z) \rightarrow \text{invited}'(f(z))(z)] \& H(f)]]$$

$$[[\text{woman}]] = \lambda f[\forall x[\text{woman}'(f(x))]]$$

There are a few ways that we could go about combining these three parts. Jacobson 2002 actually did things a little bit differently, giving as the meanings for the entire RC just the  $\langle ee, t \rangle$  meaning for each RC and then intersecting all three sets. I built the intersectivity into the RCs just to be consistent with what we did above, but note that this now leads to some trickery; nothing, I think that can't be solved with a bit of type lifting here and there, so assume that that's all in place. (What seems the simplest is to apply “that he loves” to “woman” and then shift (as you would to get a new domain restriction slot or RC slot as proposed in Jacobson 2005) to get a new argument slot for “that every man invited”.) Regardless, eventually we'd end up intersecting all three sets of  $ee, t$  to give:

$$[[\text{woman that he loves that every man invited}]] =$$

$$\lambda f[\forall x[\text{woman}'(f(x))] \& \forall z[\text{loves}'(f(z))(z)] \& \forall y[\text{man}'(y) \rightarrow \text{invited}'(f(y))(y)]]$$

$$[[\text{the woman that he loves that every man invited}]] =$$

$$\lambda G[\text{the}_{e,e}[G(h)]](\lambda f[\forall x[\text{woman}'(f(x))] \& \forall z[\text{loves}'(f(z))(z)] \& \forall y[\text{man}'(y) \rightarrow \text{invited}'(f(y))(y)]])$$

$$\text{the}_{e,e}[\forall x[\text{woman}'(h(x))] \& \forall z[\text{loves}'(h(z))(z)] \& \forall y[\text{man}'(y) \rightarrow \text{invited}'(h(y))(y)]]$$

### Binding into heads, NP-S:

$$[[\text{the woman}]] = \lambda R_{\langle ee, t \rangle}[\lambda f[\forall x[\text{woman}'(f(x))] \& R(f)]]$$

$$[[\text{that he loves}]] = \lambda h[\forall z[\text{loves}'(h(z))(z)]]$$

$$[[\text{that every man invited}]] = \lambda f[\forall z[\text{man}'(z) \rightarrow \text{invited}'(f(z))(z)]]$$

Here is where it becomes essential that we allow some way to pass up the domain restriction slot if it has been filled.

Jacobson 2005 suggests that since we need a rule in the syntax for RCs anyways, let's just make this a unary shift rule. (Although, in Jacobson 2005, this was a rule to shift N to N/RC.)

So let's just say that NPs start off with their “regular” (or unrestricted) meanings and shift to take a domain restriction or RC.

NP -> NP/RC

(I think this will work with the GQs because they will have other open argument slots to tamper with, but in the case of the definite determiner, once we close off the  $f$ , it's closed off for good. So we'll need something a little different.)

So let's say that NPs start off with a domain restriction slot (passed up, of course, by either the determiner or the noun), but can shift to take extra restrictions.

Given an NP  $\alpha$  with the meaning  $[[\alpha]]$ , there is an NP/RC with meaning  $\lambda P[\lambda Q[[\alpha]](P \cap Q)]$ .

(The types will depend on the type of the NP, of course.)

$$\begin{aligned} [[\text{the woman}]] &= \lambda R_{\langle ee, t \rangle}[\text{tf}[\forall x[\text{woman}'(f(x))] \& R(f)]] \Rightarrow_{\text{NP/RC-shift}} \\ &= \lambda P[\lambda Q[\lambda R_{\langle ee, t \rangle}[\text{tf}[\forall x[\text{woman}'(f(x))] \& R(f)]](P \cap Q)]] \\ &= \lambda P[\lambda Q[\text{tf}[\forall x[\text{woman}'(f(x))] \& (P \cap Q)(f)]]] \\ &= \lambda P[\lambda Q[\text{tf}[\forall x[\text{woman}'(f(x))] \& P(f) \& Q(f)]]] \end{aligned}$$

$$\begin{aligned} [[\text{the woman who he loves}]] &= \\ &= \lambda P[\lambda Q[\text{tf}[\forall x[\text{woman}'(f(x))] \& P(f) \& Q(f)]]](\lambda h[\forall z[\text{loves}'(h(z))(z)]]]) \\ &= \lambda Q[\text{tf}[\forall x[\text{woman}'(f(x))] \& \lambda h[\forall z[\text{loves}'(h(z))(z)]](f) \& Q(f)]] \\ &= \lambda Q[\text{tf}[\forall x[\text{woman}'(f(x))] \& \forall z[\text{loves}'(f(z))(z)] \& Q(f)]] \end{aligned}$$

And here we can either shift again or just saturate the remaining argument slot:

$$\begin{aligned} [[\text{the woman who he loves that every man invited}]] &= \\ &= \lambda Q[\text{tf}[\forall x[\text{woman}'(f(x))] \& \forall z[\text{loves}'(f(z))(z)] \& Q(f)]](\lambda g[\forall z[\text{man}'(z) \rightarrow \text{invited}'(g(z))(z)]]]) \\ &= \text{tf}[\forall x[\text{woman}'(f(x))] \& \forall z[\text{loves}'(f(z))(z)] \& \lambda g[\forall z[\text{man}'(z) \rightarrow \text{invited}'(g(z))(z)]](f)]] \\ &= \text{tf}[\forall x[\text{woman}'(f(x))] \& \forall z[\text{loves}'(f(z))(z)] \& \forall z[\text{man}'(z) \rightarrow \text{invited}'(f(z))(z)]] \end{aligned}$$

An alternative solution for stacking RCs:

RC -> RC/RC, where we intersect RCs

Given RC  $\alpha$ , then it can shift to RC/RC  $\lambda P[P \cap [[\alpha]]]$

This solution is nice because we don't have to fuss around with making assumptions about NPs. It will end up giving us the same results as well.

References:

Jacobson, P. "Copular Connectivity", in Proceedings of the 4th Conference on Semantics and Linguistic Theory. Distributed by Cornell Working Papers in Linguistics, 1994.

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

Jacobson, P. "Direct Compositionality and Variable-Free Semantics: The Case of Binding into Heads", in B. Jackson (ed.), Proceedings of the 12th Conference on Semantics and Linguistic Theory, Ithaca, NY: Cornell University CLS Publications, 2002.

Jacobson, P. "Variable Free Semantics: The Case of Quantifier Domain Restrictions", Institut Jean Nicod, June 2005