

Questions to keep in mind:

- 1) Syntax of relative clauses?
 - a) syntax of the gap
 - b) syntax of the relative pronoun
 - i) regular pronoun category?
 - ii) if not, how to get pied piping?
- 2) Syntax of contextually salient domain restrictions?
- 3) Syntax of exceptive constructions? (Should it be similar to (1) and (2)?)
- 4) Domain restriction in det/quantifier or noun?
 - a) if in det/quantifier, is it just born in the lexicon or via a shift?
 - b) if shift, what is the syntax of that shift?

===

Relative pronouns are optional when the gap of a relative clause is an object gap, obligatory for subject gaps:

- (1) Every man who likes Sally.
- (2) *Every man likes Sally.
- (3) Every man who Sally likes.
- (4) Every man Sally likes.

This seems to suggest that relative pronouns are used primarily for their syntactic (and not semantic effects). In fact, in my modification of the NP-S analysis, I assumed that relative pronouns denoted, like regular pronouns, the identity function. This was independently motivated by the fact that to get an NP-S analysis, we had to build the inteselectivity into the NP itself, not relying on the relative pronoun to provide that as Steedman does. So, the category of the clausal part of the relative clause is the category of the NPs domain restriction argument. It will also be the resulting category after the relative pronoun takes its argument.

(I've abandoned the idea that relative pronouns are EXACTLY like regular pronouns like *he*. Initially, I was concerned since this is what Jacobson's pied piping treatment was hinged upon. However, remember that we are going to keep the same semantics--sort of--, so in fact, the pied piping treatment will not be lost.)

So what category do we want to give RCs?

Two views on how to do the RC composition:

Every man who Sally likes.

- (5) Function compose $\text{lift}([\text{sally}])$ and $[\text{likes}]$ (Steedman):
 $\text{lift}([\text{sally}]): \text{sally}' \rightarrow \lambda P_{et}[\text{P}(\text{sally}')] ; \text{NP} \rightarrow \text{S}/(\text{S}\backslash\text{NP})$
 $[\text{likes}]: \lambda x[\lambda y[\text{likes}'(x)(y)]] ; (\text{S}\backslash\text{NP})/\text{NP}$
 $\text{lift}([\text{sally}]) \cdot [\text{likes}]: \lambda x[\text{likes}'(x)(\text{sally}')] ; \text{S}/\text{NP}$

(6) Introduce an extraction rule (Jacobson): $(A/B)/\dots C \rightarrow (A|C)/\dots B$ with corresponding semantic permutation of arguments (slashes are variable here)

$[[likes]]: \lambda x[\lambda y[likes'(x)(y)]]; (S\NP)/NP$

extraction shift to:

$\lambda x[\lambda y[likes'(y)(x)]]; (S|NP)\NP$

extract($[[likes]]$) applies to $[[sally]]: \lambda y[likes'(y)(sally')]; S|NP$

Motivation for (2) in Jacobson (1999)—*Every man knows what his mother likes*

(By the way, if we had wanted relative pronouns to be exactly like regular pronouns NP^{NP} , we'd have no way to combine the clausal component with that category. In the function composition approach in (5), we could lift $[[who]]$ over S^{NP} but that really wouldn't get us very far because of the rightward slash on the clausal part. (6) presents even greater challenges.)

Ok, so lets try out (6). If the clausal component of the relative clause is $S|NP$, then the relative pronoun should be $(S|NP)/?$. Given the rule in (6), for relative clauses with subject gaps:

Every man who likes Sally

$[[likes]]: \lambda x[\lambda y[likes'(x)(y)]]; (S\NP)/NP$

apply to $[[Sally]]: \lambda y[likes'(sally')(y)]; S\NP$

Assuming that our extraction rule allows for B in the schema to be null, we can apply the extraction rule to $[[likes Sally]]$:

extract($[[likes Sally]]$): $\lambda y[likes'(sally')(y)]; S|NP$

Why we want to do this? If we kept $[[likes Sally]]$ as $S\NP$, then there'd be no way to generalize a category for the relative pronoun since we had $[[Sally likes]]$ be of category $S|NP$. And it seems perfectly reasonable that our gap rule should be able to apply to both subject and object extractions.

Given that, then, the category for relative pronouns must be:

(7) $(S|NP)/(S|NP)$

Perfect! This nicely matches up with our semantics, which was the identity function. (Though this is now the identity function over properties, not individuals.)

One thing that hasn't been completely accounted for:

Our initial starting off point with the subject vs. object gaps being obligatory and optional, respectively. If the clausal component of a relative clause with subject gap can shift to $S|NP$, then there's no reason why you need the relative pronoun at all.

One story to tell:

Perhaps when the would-be B is null and you're just turning a regular slash into an extraction slash, you can only do so with external motivation; in this case, it's the relative pronoun that motivating it. Without the relative pronoun, subject gaps would remain $S \backslash NP$ (the dangers of which are obvious).

Another:

There are two *whos*: $[[who_1]]$ is the object extraction *who*, corresponding to (7). $[[who_2]]$ is the subject extraction *who*, corresponding to (8):

(8) $(S|NP)/(S \backslash NP)$

Of course, this story would have to cover all relative pronouns. It seems unlikely that for each relative pronoun with the phonological form α that there should be two distinct lexical entries for purely syntactic reasons. Then again, perhaps there are languages whose relative pronoun changes according to where the gap is due to morphological case marking? German, for example:

der Tisch, der grün ist.
 the table that(nom) green is
 'The table that is green'

der Tisch, den ich dir gebe.
 the table that(acc) I you(dat) give.
 'The table that I give you'

If we assume that RCs are of category $S|NP$, then definite descriptions must be $NP/(S|NP)$ and quantified NPs $((S/(S \backslash NP))/(S|NP)$, right? Giving definite descriptions and GQs these categories imposes the condition that the domain restriction argument can only be filled by overt linguistic material. We'd like to keep that slot available to contextually salient restrictions as well, though, so we need something else.

Possible solutions:

- 1) A new set of directional slashes. $/^$ and $\backslash^$, meaning "if my argument is linguistically overt, I'll fill to my right or left, respectively; else, the argument will be supplied by context" Our NPs would then be of categories $NP/^ (S|NP)$ and $((S/(S \backslash NP))^/(S|NP)$.
- 2) Let NPs be of categories $NP/(S|NP)$ and $((S/(S \backslash NP))/(S|NP)$.
 Let them also undergo the extraction shift mentioned above:
 $(A/B)/...C \rightarrow (A|C)/...B$
 $NP/(S|NP) \rightarrow NP|(S|NP)$ (semantics remains the same)
 $((S/(S \backslash NP))/(S|NP) \rightarrow ((S|(S|NP))/(S \backslash NP)$ (switches the argument slots so that you get the VP first)

sample derivations:

<i>every dinosaur</i>	<i>who</i>	<i>simon</i>	<i>likes</i>
$((S/(S\backslash NP))/(S\backslash NP));$	$(S\backslash NP)/(S\backslash NP); \lambda P[P]$	$NP; \text{simon}'$	$(S\backslash NP)/NP;$
$\lambda R[\lambda Q[\forall x[\text{dino}'(x) \ \& \ R(x) \ \rightarrow Q(x)]]]$			$\lambda y[\lambda z[\text{likes}'(y)(z)]]$

extraction shift:

$(S\backslash NP)\backslash NP;$
 $\lambda y[\lambda z[\text{likes}'(z)(y)]]$

$S\backslash NP; \lambda z[\text{likes}'(z)(\text{simon}')]]$

$S\backslash NP; \lambda z[\text{likes}'(z)(\text{simon}')]]$

$S/(S\backslash NP); \lambda Q[\forall x[\text{dino}'(x) \ \& \ \text{likes}'(x)(\text{simon}') \ \rightarrow Q(x)]]$

With contextually salient domain restriction:

<i>every dinosaur</i>	<i>made a diorama</i>
$((S/(S\backslash NP))/(S\backslash NP));$	$(S\backslash NP); \lambda x[\text{made-diorama}'(x)]$
$\lambda R[\lambda Q[\forall x[\text{dino}'(x) \ \& \ R(x) \ \rightarrow Q(x)]]]$	

extraction shift:

$((S|(S\backslash NP))/(S\backslash NP));$
 $\lambda Q[\lambda R[\forall x[\text{dino}'(x) \ \& \ R(x) \ \rightarrow Q(x)]]]$

$S|(S\backslash NP); \quad \lambda R[\forall x[\text{dino}'(x) \ \& \ R(x) \ \rightarrow \text{made-diorama}'(x)]]$

Now, we are in a position to actually look into the NP:

First, there are two ways we could get the domain restriction, either in the noun or in the determiner/quantifier. We showed that if the domain restriction were in the noun, to get functional readings of NPs to fall out systematically, we had to assume that the domain restriction came via a shift (1). On the other hand, if the domain restriction were in the determiner/quantifier, there's really no reason to prefer getting the domain restriction via a shift (2) over having it be born in the lexical entry for the word (3). So we have three semantic theories and will need three corresponding syntactic theories.

Theory (1):

<i>every</i>	<i>dinosaur</i>
$(S/(S\backslash NP))/N$	N
$\lambda P[\lambda Q[\forall x[P(x) \rightarrow Q(x)]]]$	$\lambda y[\text{dino}'(y)]$

domain restriction shift:

 $N \rightarrow N/(S|NP)$ $\lambda y[\text{dino}'(y)] \rightarrow \lambda D[\lambda y[\text{dino}'(y) \ \& \ D(y)]]$

function compose:

 $(S/(S\backslash NP))/(S|NP)$ $\lambda D[\lambda Q[\forall x[\text{dino}'(x) \ \& \ D(x) \rightarrow Q(x)]]]$

Our shift rule in theory (1) simply has the semantics $[[N]] \rightarrow \lambda P[[[N]] \ \& \ P]$; it's an untyped generalized conjunction rule to account for functional domain restrictions as well as our ordinary ones.

Theory (2):

<i>every</i>	<i>dinosaur</i>
$(S/(S\backslash NP))/N$	N
$\lambda P[\lambda Q[\forall x[P(x) \rightarrow Q(x)]]]$	$\lambda y[\text{dino}'(y)]$

domain restriction shift:

 $(S/(S\backslash NP))/N \rightarrow ((S/(S\backslash NP))/(S|NP))/N$ $\lambda P[\lambda Q[\forall x[P(x) \rightarrow Q(x)]]] \rightarrow \lambda R[\lambda R[\lambda Q[\forall x[P(x) \ \& \ R(x) \rightarrow Q(x)]]]]$ $((S/(S\backslash NP))/(S|NP))$ $\lambda R[\lambda Q[\forall x[\text{dino}'(x) \ \& \ R(x) \rightarrow Q(x)]]]$

Theory (2), honestly, doesn't seem like something we want to go with. As seen in the other write-up, to actually get the semantics to work out, we're actually geaching the quantifier twice and then applying it to the domain restriction shift rule. It is essentially the same thing as theory (1), but with some weirder assumptions (like being able to apply the semantic value of an overt piece of linguistic material to the semantic effect of an unpronounced unary shift rule).

Theory (3):

<i>every</i>	<i>dinosaur</i>
$((S/(S\backslash NP))/(S NP))/N$	N
$\lambda P[\lambda R[\lambda Q[\forall x[P(x) \ \& \ R(x) \rightarrow Q(x)]]]]$	$\lambda y[\text{dino}'(y)]$

$((S/(S|NP))/(S|NP))$
 $\lambda R[\lambda Q[\forall x[\text{dino}'(x) \ \& \ R(x) \ \rightarrow \ Q(x)]]]$

Back to the category of RCs and domain restrictions:

Ultimately, I'm unsure that we want to say that all domain restrictions are S|NP, or even that relative clauses are of type S|NP. We'd like for these things to have the same category as exceptive constructions, since they do have similar distributions (the limited distribution of exceptives has more to do with its semantics than syntax, I think). Furthermore, consider question-marking wh-words, as in:

Every student knows what dinosaurs like to eat.

since this particular use of *know* subcategorizes for a question (category Q), *what* is of category Q/(S|NP). A nice parallel to this approach to question-marking wh-words would be to say that relativizing wh-words are of category R/(S|NP), where R has the semantic type e,t. The advantage to giving a more abstract category is that now we are not committed to say that all domain restrictions are S|NP or relative clauses. Of course, we don't want to say that *any* e,t type material can be a domain restriction (we don't want *dog* to be a possible contextually salient domain restriction to *every dinosaur*), so we'll need something extra to ensure that the right things become Rs. Not sure how reasonable it is to assume that contextually salient properties themselves, in order to become candidate domain restrictions, go through a R/X shift to become Rs, but if that's at all reasonable (there has to be *some* process by which things become contextually salient, right?), then we could say that only PPs, S|NPs, etc. can go through this shift. The sort of attractive bonus to calling domain restrictions S|NP is that pretty much any domain restriction can be phrased as a relative clause.