A Categorial Grammar View of Syntactic Categories:
Some Unexpected Evidence from Coordination

Pauline Jacobson
Linguistics Program
Brown University
pauline_jacobson@brown.edu

Draft, Nov. 18 - Comments Welcome
(in submission)

1. Introduction

The goal of this paper is to show that under a rather natural view of syntactic categories compatible with the general theory of Categorial Grammar (hereafter, CG), some rather surprising and hitherto undiscussed (to my knowledge) facts about coordination fall out. It is standard in CG to maintain that there is a small set of primitive categories and a recursive definition of others; these others are often referred to as 'function' categories (not to be confused with the notion of 'functional categories' in other theories of syntax). While expressions of one of these function categories have meanings which are functions, the use of the term 'function' in the syntax is - at least in some of the CG literature - metaphorical. Here, however, we advocate a view in which each such category literally corresponds (oversimplifying for the moment) to a mapping from strings to strings. We will see that by taking the notion of functions in this sense seriously, we limit the ways in which strings can combine which in turn makes some striking predictions.

To give a spoiler alert: the general view advocated here effortlessly handles the existence of surprising sentences like (1):

1. a. Barry plays the dulcimer, the standup bass, and (he) does magic tricks.

   b. Sarah studies syntax in the morning, semantics in the afternoon, and (she) practices the violin in the evening .

   c. Barry plays the dulcimer, the standup bass, does magic tricks, and he builds treehouses. .

Sentences of the form in (1b) were first noted in Maxwell and Manning (1996) but have received little attention, and (1a) is a simplified version making the same point. It is surprising to see an ordinary object coordinated with a full VP or even a full S; (1b) has the extra fanciness of including an instance of 'nonconsitutent coordination'. And note that we can construct cases like where the last S doesn't even have the same subject:

2. Barry plays hammered dulcimer, the bass, does magic tricks, and Patty plays the flute.

But an additional surprise not noticed by Manning and Maxwell (who did not explore the domain in any detail) is the fact that when there is a chain of such coordinations, the 'bigger' ones can only be on the right. Thus notice that in all the above cases the constituents 'grow' as we move right. For example, (1c) and (2) are of the form NP, NP, VP, and S.[1] But we cannot mix these to give the order VP, VP and NP for example:

3. *Barry plays the dulcimer, does magic tricks, and the standup bass.

Sec. 5.2 develops additional examples.

Thus the twin punchlines of this paper are: (1) the existence of the Maxwell/Manning type of cases is unsurprising, and (2) the 'grow only rightward' phenomenon also follows from the view of syntactic categories here. We also briefly turn to the placement of *either* discussed in Larson, 1985, Munn 1993, Schwarz 1999, and den Dikken 2006, and in particular what the latter called 'too low' *either*. I show that this also falls out from the analysis here (it is a special case of the Maxwell/Manning phenomenon) although the account leaves open questions about the full set of facts regarding *either*.

Sec. 2 of this paper elucidates the background. Nothing in that section is new in this paper; most is just based on the framework outlined in Jacobson (2014). However, since the reader of this paper might not have a background in CG, I will supply here the necessary ingredients. (It should be kept in mind that none of these ingredients were designed for the data in this paper.). Sec. 3 discusses the status of the Coordinate Structure Constraint - arguing with many others (e.g., Lakoff, 1986, Kuno, 1987, Kehler 1996) that this is not a syntactic constraint. There we also develop an account of Null conjunction (as in *Peter, Paul, and Mary are all musicians*). Sec. 4 turns to the interaction of null conjunction with Right Node Raising. The major results of the paper are in Sec. 5 where we turn from the situation with respect to Right Node Raising to what we might call 'Left Node Raising', i.e. to the Maxwell/Manning cases and the 'grow rightward' generalization. Sec. 6 briefly turns to *either*.

2. Background Pieces

2.1. Some basics of Categorial Grammar

We begin with some basics of the syntax (and semantics) of Categorial Grammar (CG). While there actually are many varieties of CG and related work in the Type Logical tradition, I base most of this on a version elucidated in Jacobson 2014. This is close in some ways to what is known as Combinatory Categorial Grammar (Steedman, 1987, to appear, and many papers in between) but departs from that in significant ways, some of which will be noted as we go. The first point here is hardly unique to CG: we take any linguistic expression (of any size) to be a triple of <[sound] (represented here with orthography), Category, meaning>. By meaning we mean a model-theoretic object – not a representation in a symbolic language, although such representations are used as convenient ways to *name* model theoretic objects. One might wonder where in this view is encoded the internal structure of an expression - i.e., its tree representation. Under the particular view here a tree is but a representation of how an expression is put together and plays no role in the grammar itself. That is, the grammar does not 'see' trees. (One caveat: we explore below the possibility that a small bit of internal structure does need to be kept track of.) Second, we can think of syntactic categories as being basically just encodings of an expression's distribution; in most versions of CG (including the version here) the category also encodes the type of meaning that an expression has. Thus there are a few primitive categories, and a recursive definition of all others. For the primitives, we begin with the set {NP, S, N, PP, CP}; there could be a few others. Also, as is quite standard, there is a set of features that can occur on these categories. Other categories are defined recursively; as a first pass (to be refined momentarily) we can say that if A is a category and B is a category than A/B is a category. The interpretation of these categories is that an expression of category A/B is something that wants to combine with an expression of category B to give as a result an expression of category A. Of relevance to some of the secondary results here is the question of whether features should always be on (or part of) some basic category, or whether a feature can 'span' a category of the for A/B (or a more complex category). It will turn out both to simplify how features pass and to have some added bonuses (see the discussion surrounding (47b) and see Sec. 6) if we assume that in the cases where we might intuitively think of a feature as 'spanning' a category, it is actually encoded on the result. (Notice that if we think of even the so-called 'basic' categories like NP, S, etc. as actually being features - a view first put forth in Chomsky 1970) - and all non-function categories being feature bundles, it makes no sense in any case to think of the features as spanning function categories; they are always part of some non-function category.) However, wherever we aren't spelling out a full category and using some label "X" as an abbreviation of a complex category, we write X[F] for convenience, with the understanding that the feature F is actually on the ultimate result category.

Note that the above doesn't specify how an A/B combines with a B. We therefore refine this to have the following recursive specifications of categories: If A is a category and B is a category then $A/_RB$ is a category and $A/_LB$ is a category. The intuition behind these categories is that each encodes how the relevant expression will combine with its 'argument'. An expression of category $A/_RB$ takes a B-expression to its right to give as result an expression of category A, and an expression of category $A/_LB$ takes its argument to the left. (Other authors within the CG and related traditions use other notations.)

Before continuing, a note about the semantic types and semantic combinatorics. We assume that all expressions of some primitive category A have the same type of meaning - hence each such expression denotes a member of some set, call it a. Any NP denotes something in the set e - i.e., it denotes an individual (this means that generalized quantifiers like *every duck* are not NPs, see Sec. 2.5.1.). All Ss denote something of type t (a truth value). Notice that we are dealing here only with extensions - this is for expository convenience only. Further, any expression of category A/B has as its extension a function from the type of meanings that B expressions have to the type that A expressions have; thus a function from set b to set a, the set of such functions is notated here (as is standard) as <b,a>. Thus to say that something has as its "semantic type <b,a>" means that its extension is a function in that set. Thus *walks,* for example has as its category $S/_{L}NP$ and a meaning of type <e,t>. A transitive verb *likes* is of category $(S/_{L}NP)/_{R}NP$ and thus has as its meaning some function in the <e,<e,t>>. Unsurprisingly, the semantics will ensure that when an expression of category A/B combines with one of category B, the associated semantics is functional application. I will often use "VP" as an abbreviation for $S/_{L}NP$, and TV as an abbreviation for $(S/_{L}NP)/_{R}NP$. To make completely explicit how the syntactic and semantic combinatorics works we posit that the grammar contains two rule schemas as follows:

(4). a. Given an expression α of the form <[α]; $A/_{R}B$; [[α]]> and an expression β of the form <[β]; B; [[β]]>, there is an expression of the form <[αβ], A, [[α]]([[β]])>.

b. Given an expression a of the form <[α], $A/_{L}B$, [[α]]> and an expression β of the form <[β], B, [[β]]>, there is an expression of the form <[βα], A, [[α]]([[β]])>.

The two schemas in (4) look surprisingly alike – the only difference is in the directionality of the two expressions – suggesting that they should be collapsed. Of course, if we are just talking about having one collapsed rule vs. two rules, the difference would seem to be negligible. But if there are additional operations such as the case of a function taking its argument as an infix (what Bach 1979 dubbed "Wrap") or the reverse - where the function itself is an infix, then there would be more rules to collapse. Moreover, the bulk of this paper will show that the tool needed to collapse these is exactly what leads to the main predictions elucidated later.

In order to collapse the two, the idea is to take seriously the notion of syntactic categories as corresponding to functions - specifically, functions on strings.[2] Indeed the CG literature often speaks of the 'function' category but in some works that is either a metaphor or just means that the meaning of an expression of the relevant category is a function.[3] But let us take literally the idea that these categories correspond to functions. I say 'correspond to' rather than 'are' functions because the primitive categories cannot be functions; there we can take the label "NP", for example, to denote a set of strings. In view of this, a category of the form A/B should also be taken as denoting a set of strings (rather than a function). But we can take any such category as corresponding to some function. Mirroring the semantics, the first stab at this would be to say that the function corresponding to $S/_L NP$ is a function from NP strings to S strings. This will not quite do - it only makes sense to think of this as mapping an NP string (say, e.g., *Lee*) to an S string likes *Lee walks* after we have already 'plugged in' *walks.* This means that the function corresponding to $S/_L NP$ first has to apply to an expression of that category. Technically, then, there is a function corresponding to $S/_L NP$ which is a function in <string, <string, string>>. It applies to an expression of that category (e.g., *walks*) to give the desired function from strings to strings. Once that function applies to *walks* we then have the function mapping *Gerald* to *Gerald walks, Sally* to *Sally walks*, etc. For every category X, we refer to the corresponding function as F:X. (read that as the function corresponding to the category X), and for every function F we refer to the corresponding category as CAT:F. We then define the relevant functions as follows:

(5) a. $F{:}A/_{R}B([x])([y]) = [xy]$ b. $F{:}A/_{L}B([x])([y]) = [yx]$

With this we collapse (4) into a single schema as follows (where X ranges over R and L):

(6) Given an expression α of the form $<[\alpha], A/_{X}B, [[\alpha]]>$ and an expression β of the form $<[\beta], B, [[\beta]]>$, there is an expression of the form $<[F{:}CAT_{\alpha}([\alpha])([\beta])], A, [[\alpha]]([[\beta]])>$.

2.2. Word order rules

The ultimate word order in a sentence is determined by the categories of the lexical items, but obviously these are not listed on a case-by-case basis. For example, we clearly don't what every intransitive verb in English to just happen to be listed in the lexicon with category $S/_{L}NP$ as opposed to $S/_{R}NP$. To this end, Jacobson 2014 suggests that lexical items are (in general) 'born' in an underspecified form and thus without directional slashes, and those are added by general rules. A first pass at these rules would be as follows (from Jacobson 2014); the .... on either side of the category means that these rules apply even when that category is embedded in others. The intent also is that the directional slashes are only added where there are none; no rule overrides already assigned directional features:

(7) a. ...(S/X)... $\rightarrow$ ... $(S/_{L}X)$...

b. ...(X/X)... $\rightarrow$ ...$(X/_{L}X)$...

c. The default rule: ...(A/B) $\rightarrow$ …$(A/_{R}B)$…

Being the default, the rule in (c) has to apply after (a) and (b). Interestingly, these are also reminiscent of the word order parts of the three standard X-bar rule schemas as spelled out for English. There is no formal notion of a specifier here (nor, in fact, of a head) but (a) ensures that 'subjects' appear on the left. (b) says that modifiers are also on the left (this is obviously much too general for English since there is a certain degree of freedom in the placement of modifiers but it will serve for a first pass), and (c) says that otherwise things take their arguments to the right, much like the 'head first' generalization.

2.3. A brief digression on infixation

What about 3-place verbs as in *give the bone to Fido*? Space precludes a full discussion, but some remarks are important as the treatment of these interacts with the material in Sec. 4. Following a long tradition begun originally in Chomsky 1957 and then generalized and extended (using various formalisms) in, e.g. Bach 1979, 1980, Dowty 1982 (both of those within CG); Pollard 1984 within an extension of GPSG which later because HPSG: Jacobson 1987 within GPSG, and Larson 1988 within a movement based theory - it is quite possible that *give* first combines with *to Fido* and that subsequently *the bone* is introduced. Rather than using movement to account for the fact that the surface order contains a 'discontinuous constituent', Bach proposed that *the bone* is directly introduced as an infix, and referred to this as Wrap. If this is correct, we need another directional feature - call it I - indicating the argument is taken as an infix. Thus *give* would have the category $((S/_{L}NP)/_{I}NP)/_{R}PP$. This also means that when *give to Fido* combines with *the bone,* the former is not just a string, but needs a specified infixation point. Also needed are conventions as to whose infixation point is inherited in the new expression; we assume in general this is inherited from the function category. And in a full system with infixation we will also want a 'circumfix slash' ($A/_{C}B$) indicating that the expression with this category is itself an infix - i.e., it takes its argument as a circumfix. This would be needed once we introduce Lift (Sec. 2.5.1).

An important question for the material in Sec. 4 is the following. If there are infixation slashes, then after the word order rules apply, what is the actual category of a simple transitive verb like *likes?* The most general statement of the rule adding a I-feature to a slash is for the rule to map any item of category ((S/X)/NP)/... to $((S/X)/_I NP)/...$ That means that even a simple transitive verb takes its object as an 'infix', but in the case where there is no other argument, the infixation is vacuous (the infixation point in English is presumably on the right side of the verb, and so adding the object to the right of the infixation point yields the same result as adding it after the verb). That said, the remarks below are greatly simplified if we continue to treat an ordinary transitive verb as being of category $(S/_L NP)/_R NP$ - which is what we would get if the infix slash is only specified on items of category ((S/X)/NP)/Y (i.e., on 3-place verbs). Since this is a (very) slight complication in the statement of the word order rules, one might wonder whether treating transitive verbs as being marked with R rather than I on the object position is justified.

I believe it is. First, it is certainly not a major complication to restrict the infixation rule to only 3 place verbs, such that the rule applies takes as input only items of the category ((S/X)/NP)/Y where Y is not null. Second, everything of category $((S/_L X)/_I NP)/...$ also has a version where the I slash is an R slash - this is needed for the case of "Heavy NP Shift", and so there is really no harm in working with the R-slash version of transitive verbs. ("Heavy NP Shift" in those cases is vacuous.). Third, even leaving aside the Heavy NP Shift version of the category and even if we do make the assumption that ordinary transitive verbs take their objects by vacuous infixation, I strongly suspect everything being said in Secs. 4 using our simplification would carry over to a more fully articulated theory with infix (and circumfix) slashes. But since we are not articulating a full theory with infixation, for now we will for the most part set aside the project of looking in detail at infixation more generally. And, finally, while this might ultimately mean that the discussion in Sec. 4 needs revision, the twin 'punchlines' - which are developed in Sec. 5- are not affected as they are entirely about L slashes.

2.4. *and* and *or*

We turn next to the category for *and* (and *or*). We assume that *and* is listed in the lexicon as a cross-categorial connective and thus of category (X/X)/X, with the generalized semantics given in Parte3 and Rooth 1983. (We slightly refine its syntactic category in Sec. 3.) Note that the proposed directional features predict that this will be fleshed out as $(X/_{L}X)/_{R}X$. There is, of course, nothing novel about this prediction; most theories about word order principles will make the analogous prediction. See, e.g, Munn 1993.

2.5. Additional combinatory rules and applications to conjunction

2.5.1. Lift

Consider first the syntactic category of generalized quantifiers like *every dog.* Under the view of the syntax/semantics match here, these cannot be NPs as they do not denote individuals. Using the fairly standard wisdom since Montague 1973, their semantic type is <<e,t>,t>. In subject position they take the VP as argument which leads us to the conclusion that their syntactic category is $S/_{R}(S/_{L}NP)$, and a quantificational determiner like *every* is of category $(S/_{R}(S/_{L}NP))/_{R}N$. (This says nothing about generalized quantifiers in object position; see Hendriks 1993 and Jacobson 2014 for relevant discussion). But note an anomaly about this category: the directional slashes are not expected. The L-slash is not surprising and follows from the word order rules given in (7), and the directional feature on the final slash in *every* is the unsurprising default R feature. What is surprising is the first (reading left to right) R slash. Should this not be an L-slash, since in general whatever combines with any X to give an S takes that S to the left? We return to this directly.

Partee and Rooth 1983 proposed that ordinary NP meanings (individuals) can also lift to generalized quantifier meanings (type <<e,t>,t>) ("Montague's meaning" in Montague 1973) such that if the name *Lee* denotes the individual l, then the lifted meaning of *Lee* is $\lambda P[P(l)]$ – i.e. function mapping any <e,t> function P to the value that P assigns to l to (in set terms, the set of sets containing l). This was motivated by the fact that *Lee*, for example, can conjoin with a generalized quantifier (*Lee and every candidate for schoolboard will speak at the meeting*). But rather than simply list the meaning of *Lee* in the lexicon as the 'fancier' meaning (as Montague did), Partee and Rooth argued that it is 'born' with the simpler meaning (of type e). We will adopt the Partee and Rooth semantics, but given the rest of the premises of CG we assume that the lift rule also maps the syntactic category NP to the generalized quantifier category $S/_R(S/_L NP)$.

Moreover, we will assume that the rule is much more general (for an early application generalizing this see Dowty, 1988) and allow any expression to lift. By way of formalizing this, notice that lift is an operation which is defined perfectly generally. Take any set A, any member a of this set, and any set of functions <A,B> (i.e., the set of functions in AXB). Then we define (Lift(a)) as that function taking as argument any function f in

<A,B> and mapping f to that value that f assigns to a. The result is thus something in B, so Lift(a) is of type <<A,B>,B>. We need to be a bit more precise here: Lift(a) does not yield a unique function because it depends what set we are lifting 'over'; hence when needed we will write Lift$_{<A,B>}$(a) to mean the function whose domain is the set of functions <A,B>. (It follows by definition that the codomain is the set B.) Restating the idea of Lift informally so as to make the intuition clear, we start with some object a. The lifted a is now the 'boss', it takes some function f (that could have taken a as argument), and gives back just what would have resulted had f applied a.

Given the rest of this version of CG, we also want to tie this in to the syntax. Hence we can write a rule mapping a single triple into a new one, which differs in meaning and category but not in sound. This is an instance of what is often called a type shift rule, but we avoid that terminology here since it also affects the syntactic category. Note further that there can be no notion of 'type shift as a last resort' in the architecture of the grammar assumed here. That makes sense only if the syntax precomputes representations which are 'sent' to the semantics whose job it is to compositionally interpret these representations (and so if it encounters a type mismatch, it has at its disposal a set of type shift rules to 'repair' these mismatches). Here the syntax and semantics are computed in tandem so these kinds of notions are inapplicable. The general lift rule is (preliminarily) as given in (8). Note that because there are two possible outputs in terms of the syntactic category, this is actually two rules which we have simply exhibited together:

(8) Given a linguistic expression of the form <[α], A, [[α]]>, there is an expression β of the form <[α], B/$_L$(B/$_R$A) or B/$_R$(B/$_L$A), [[lift$_{<a,b>}$(α)]]>.

The only difference between the two rules in (8) is the syntax of the new category. Aside from the discomfort of really having two lift rules packed in here, this looks pretty suspicious for another reason: why in both cases does the new category just happen to preserve word order? For example, when *Lee* lifts over VPs, why is the result of combining lifted *Lee* with *walks* exactly the same as *walks* combines with unlifted *Lee?* Is it just a stipulation that there is no lift rule yielding, for example, $B/_L(B/_LA)$ as the new category?

The reader has probably anticipated the answer: if we take seriously the notion of syntactic categories as corresponding to functions then the word order preservation property of Lift is no accident. By definition, a lifted object takes some function as argument to give exactly what would have resulted had that function taken the original object as argument. Hence we merely need to extend this notion to the syntactic categories. The translation into categories is, unfortunately, not immediate for two reasons. First the categories themselves are not functions but correspond to functions, Second, there is the extra layer that the function first needs to apply to the string (sound) part of the expression in order to be a function of the right type. But we can nonetheless extend the basic idea of lift to syntactic categories as follows. First we define F'-α as that function in <string,string> obtained by applying $F{:}CAT_\alpha([\alpha])$. In prose, F'-α is that function from strings to strings that results from applying the function corresponding to the category of α to the string $[\alpha]$. Since the syntactic part of the lift rule maps a category A to something that takes expressions of category B/A as argument, we will notate that as $Lift_{A/B.}$ With this we can see the sense in which the syntactic lift is the same as the general lift operation:

(9) Given any category A, $Lift_{<A/B>}(A)$ is that category C such that for all expressions α such that [α] is in A and all expressions β such that [β] is in (i.e. A/B), F'-α([β]) = F'-β([α]).

In other words, the new category C is such that for any string [x] in A and any string [y] in A/B, applying the function corresponding to C to [x] is a function mapping [y] to the string which is the string resulting from applying F:A/B to [y] and the to [x]. For example, since F-*walks* maps [Lee] to [Lee walks], then the lifted category (CAT) of *Lee* is such that F:CAT applied to [Lee] maps [walks] to [Lee walks]. From this, it follows that the lifted category can take an expression of category B/A to its right only if that expression actually had a category looking for an A to its left, and vice versa. It is thus not a 'bug' that lift happens to preserve order; it follows from the general notion of lift extended to categories. Note that if we have infixation slashes and corresponding Wrap (circumfix) slashes those would immediately fold in to the system as well: A could lift to $B/_C(B/_IA)$ or to $B/_I(B/_CA)$.

We can now return to the earlier puzzle of why a quantificational determiner has the surprising category $(S/_R(S/_LNP))/_RN$. Recall that most of those directional features are predictable, but the first (in the left-to-right sequence) R-slash is surprising. Since the last argument in in forming an S is usually taken to the left, why would the VP be taken to the right here? Similarly, in the case of a lexical generalized quantifier like *nobody* why is its category $S/_R(S/_LNP)$? The answer for the case of *nobody* is that this is simply listed with category $Lift_{S/NP}(NP)$. It follows from that that it can be $S/_R(S/_LNP)$. It could also be $S/_L(S/_RNP)$. Similarly, a quantificational determiner such as *every* is listed in the lexicon with category $Lift_{S/NP}(NP)/N$. Incidentally it doesn't matter that there are items listed as being of category $Lift_{S/NP}(NP)$ which don't have identical strings of category NP (i.e., the generalized quantifiers), since the category $Lift_{S/NP}(NP)$ is still perfectly well defined.

2.5.2. Function Composition (or it's Curry'ed version - "Geach")

Steedman (1987 and many subsequent works) proposed adding function composition as a possible combinatory rule, and showed how this can account for leftward *'wh'*- extraction cases. As will be detailed in Sec. 2.6 I will actually treat leftward (*wh*-) extraction in a slightly different way; one new rationale for is given in Sec. 4. Nonetheless, having function composition (or a variant thereof) has other striking payoffs especially with respect to coordination. The general operation of function composition is defined as follows. Given a function f in <a,b> and a function g of type <b,c>, then g $\circ$ f (read "g compose f") is a function in <a,c> and is $\lambda x_a[g(f(x))]$. Initially then we can add two general rules to the combinatory apparatus (to be revised below):

(9) a. Given an expression α of the form <[α], $A/_RB$, [[α]]> and an expression β of the form <[β], $B/_RC$, [[β]]>, there is an expression of the form <[αβ], $A/_RC$, [[β]] $\circ$ [[α]]>. ("right composition")

b. Given an expression α of the form <[α], $A/_LB$, [[α]]> and an expression β of the form <[β], $B/_LC$, [[β]]>, there is an expression of the form <[βα], $A/_LC$, [[β]] $\circ$ [[α]]>. ("left composition")

With these schemas, note that a sentence like *every candidate tells lies* has two possible derivations which result in the same word order and the same meaning. The familiar derivation is the one in which *tell* (of category $(S/_LNP)/_RNP$) first combines with the NP *lies*. This yields *tells lies* of category $S/_LNP$, which can then be taken as argument of *every candidate* of category $S/_R(S/_LNP)$ with the familiar semantics. Alternatively, *every candidate* can function compose with *tell* as the reader can easily verify to give the $S/_RNP$ *every candidate tells.* When this takes *lies* as argument we get the same meaning as the 'traditional' derivation. Indeed this follows by the very definition of function composition: $g_of(x) = g(f(x))$, and hence [[every candidate]]$_o$[[tells]] ([[lies]]) = [[every candidate]]([[tells]]([[lies]])).

The same possibilities are there with an ordinary subject like *Lee*. Here the minimal derivation is the ordinary one where *tells* combines with *lies,* and *tells lies* takes as argument the NP *Lee.* But *Lee* can also lift to the generalized quantifier category, in which case *Lee tells lies* has the two derivations exactly analogous to the case of *every candidate tells lies.*[4] In particular there is an analysis by which *Lee tells* is a well-formed expression. (See Jacobson 2014 for an additional way of allowing *Lee tells* to be a well-formed expression. This involves defining all of the rules recursively and 'holding off' on argument slots, which in turn allows *Lee* as NP to directly combine with *tells* without first lifting; the result is also $S/_RNP$ with meaning λx[tells'(x)(Lee')]. This simplifies some of the derivations, but as this possibility is not crucial to any of the material here, we will instead, when needed, use the more familiar lift+composition derivation.)

One benefit of these 'extra' and unusual constituent structure analyses is that the existence of all sorts of cases of so-called non-constituent coordination and/or Right Node Raising follow effortlessly. Thus a typical "Right Node Raising" example like (10) is simply the coordination of two expressions of category $S/_R NP$ which then ultimately take the NP *model theoretic semantics* as argument:[5]

(10) Lee loves and Sandy hates model theoretic semantics.

Dowty 1988 showed how lift and function composition together allow for a constituent like *lobster on Tuesdays* in (11), which in turn predicts the existence of surprising so-called non-constituent conjunction cases like (11)

(11) Captain Jack serves scallops on Mondays and lobster on Tuesday.

On Monday is a VP modifier, so of category $VP/_L VP$. *Scallops* is an NP, but can lift over transitive verbs to be of category $VP/_L TV$. These two can combine by the left composition rule in (9b) yielding *scallops on Monday* of category $VP/_L TV$. In other words, this (like lifted *scallops* itself) wants a transitive verb to its left to give a VP. *lobster on Tuesday* is composed in the same way, the two conjoin, and take the transitive verb *serves* to the left.[6]

The claim that *Lee loves* can be a constituent within *Lee tells lies* is of course often met with shock, and a suspicion that anyone making such a claim must have never taken a beginning syntax course. After all, don't we learn in baby syntax that there is a VP category (the particular name is not relevant) and hence *tells lies* is a constituent in *Lee tells lies?* (Recall that here we are taking 'structure' just as a way to represent how the rules prove something well formed, but no harm will be done by using the standard terminology of 'constituent structure'). Moreover, both *Lee tells* and Dowty type cases like *lobsters on Tuesday* in (11) pass none of the constituent structure tests except coordination. In fact, though, there are several reasons that this common reaction is unwarranted. The first one is based on an unstated assumption that is almost never made overt. The arguments showing that *tells lies* can be a constituent in *Lee tells lies* (and that thererefore there is a VP category) are just that - they show that the standard view of the 'structure' of this sentence is one possible structure. In no way does it follow from that this this is the only structure. This fact is almost never noted in introductory discussions - and the leap from there being a VP constituent to the conclusion that *Lee tells lies* has only one possible structure seems to be based on the hidden but unwarranted assumption that an unambiguous sentence has only one structure. Not only does this assumptiosn follow from nothing, but actually under any theory it is almost certainly incorrect: it would take a lot of extra work to preclude two different stuctures for, e.g., *Roses are red and violets are blue and irises ae white.* The two structures correspond to the same meaning. As to the fact that the 'strange' extra structures pass none of the textbook constituent structure tests except coordination, that is also true for many other well accepted constituents. The italicized expressions in (12) also pass none or almost none of the standard tests:

(12) a. Lee liked the meal *which that restaurant served.*

b. Lee believes that *the earth is round.*

c. Lee believes that Sally *walked to the bus stop.*

Indeed, is well known that the familiar battery of tests are all one-way tests: to pass is to be a constituent, to fail shows nothing. Much more than simple constituency is needed for each of these tests, and once one probes deeper into some of them it becomes quite unsurprising in some cases that expressions like *Lee likes* does not pass a particular test. Take, for example, 'substitution by a proform'. Arguably, this can be recast to say that a certain category has a lexical proform of that category. Assume, for the sake of argument, that *do so* is a VP 'lexical anaphor' - i.e.., a lexical anaphor of category $S/_{L}NP$. (This is almost certainly incorrect; *do* is a main verb and *so* is an anaphor with a somewhat complex distribution; see, e.g., Miller 1990 for discussion. But simply for the sake of exposition we will here - at the risk of propogating what is surely a false view - go along with the common introductory textbook view that *do so* itself is the 'proform'.) Then we might ask, why is there no lexical anaphor of category $S/_{R}NP$ (which is, under the CG analysis here, the category for *Lee loves*)? The answer is simple: any item of category S/NP listed in the lexicon will be assigned L and not R on the slash by the general word order rules. See Jacobson 2014 for discussion of this and some of the other 'tests'. The only phenomenon which seems to require (almost) nothing but 'constituency' is coordination, which follows if *and* and *or* are listed as (X/X)/X, for X a variable over categories.[7]

Let us return now to the two function composition schemas in (9). These raise two questions. First, should they not be collapsed? Second, is it an accident that we have just these two? For example, why is it that when $A/_{R}B$ combines with $B/_{R}C$ the result is $A/_{R}C$ and not $A/_{L}C$? That particular question is answered in Steedman (to appear) who takes the directional features to actually be features on the 'argument', and so if one were to compose $A/_{R}B$ with $B/_{R}C$ the result would have to have an R feature (as that feature is on C). But one still needs an additional principle to ensure that the direction of combination is the one dictated by what Steedman refers to as the 'principle' functor (in this case $A/_{R}B$). (Steedman's Principle of Consistency does that, but it needs to be stated as an extra principle.) And even if one succeeds in ruling out the case above, there remain other possibilities. Could there be another rule just like those in (9a) and (9b) but where the syntax is such than an expression of category. $A/_R B$ to take to its right a $B/_L C$ to give $A/_L B$? This is sometimes called 'mixed composition' (or "Crossing composition') and is, indeed, allowed in Steedman (1987, to appear). But this raises new problems. This, for example, would allow *believe* (of category $VP/_R S$) to take *wash the dishes* (category $S/_L NP$) to its right to give *believe washed the dishes* whose category would be $VP/_L NP$ to give a well-formed VP like *Sally believed washed the dishes* and thus ultimately sentences like **Lee Sally believed washed the dishes* (meaning that Lee believed that Sally washed the dishes). Incidentally, this is not to say that it is inconceivable that *believed washed the dishes* can combine - as is well know, this is found, for example, in leftward *wh-* type extraction cases (we return in Sec. 2.6). But while this may be a well-formed expression, we don't want this expression to have the category $VP/_L NP$ which is what would be allowed if mixed composition was possible.[8] But why is this type of "composition" precluded? Put differently, why are the only instances of syntactic function composition (as demonstrated by coordination possibilities) just those which preserve word order?

Once again, the reader will likely have anticipated the answer. If categories correspond to actual functions in <string,<string,string>> and 'function composition' as literally the composition of two functions (resulting after the function-category is applied to the string in question), then the schemas in (9) are the only two possibilities. A caveat here is that if there are C and I slashes then there would be additional possibilities, but the absence of the 'mixed' composition case discussed above is not affected by that possibility. The absence of this kind of composition is at the heart of the results discussed in Sec. 5, and so the material there can be seen as additional evidence against mixed composition. And this, in turn, supports the view of syntactic categories advocated here.

Thus we can reformulate (9). Recall that for any α of category X, F'-(α) is that function which results F:X is applied to [α], hence (9a and b) can be recast as (13):

(13) Given an expression α of the form <[α], A/B, [[α]]> and an expression β of the form <[β], B/C, [[β]]>, there is an expression γ of the form <F'-α ∘ F'- β ([α])([β]), CAT of γ is such that $F'\text{-}(\gamma) = F'\text{-}(\alpha) \circ F'(\beta)$, [[α]] ∘ [[β]]>.

2.5.3. Recasting Function Composition as the "Geach" rule

Function Composition is a binary operator: it takes two functions as input and yields a third as output. But it easily be recast instead by Curry'ing the function composition operator such that it takes its arguments one at time. The unary version of function composition is often referred to in the CG literature as the "Geach" rule (or in some literature "Division"), which we will notate as **g**. This rule thus takes a function in <a,b> and maps it to a function in <<c,a>,<c,b>> such that for any function h in <a,b>, $\mathbf{g}(h) = \lambda X_{<c,a>}[\lambda C_c[h(X(C))]]$. The reader can verify that for any h of type <a,b> and any f of type <b,c>, $\mathbf{g}(h)(f) = h \circ f$. (Note that **g** is not a single operation on functions because it depends what is the type of the newly introduced argument slot, so technically the case above should be notated as $\mathbf{g_c}$, see Jacobson 2014 for details.) Jacobson 2014 provides evidence for using this rather than function composition; the main difference is that using the "Geach" rule introduces an additional step in the composition where the result of the **g** might input other things. Another advantage of Geach plus application is found in Sec. 5.1

Such a modification, however, does not change any of the remarks in this paper. If we recast function composition as instead the Geach rule, then we can see that the only two ways to apply this to syntactic categories (leaving aside I and C slashes) are to map an $A/_RB$ to an $(A/_RC)/_R(B/_RC)$ and an $A/_LB$ to an $(A/_LC)/_L(B/_LC)$ - these are the only applications of a rule which we will call **g** such that for any two functions h and f, **g**(h)(f) = g o f. We will call the mapping of $A/_RB$ to $(A/_RC)/_R(B/_RC)$ the "Geach equivalent" of right composition, and similarly for the left version. Of particular relevance, we do not allow an $A/_RB$ to map by g to $(A/_LC)/_R(B/_LC)$, for that would be the Geach equivalent of one type of mixed composition. Again the view here claims that this is not ruled out by stipulation, but by the definition of the Geach operation combined with the notion that a category corresponds to a function which - once applied to a string of that category - is a function in <string,string>. And the mapping of, for example $A/_RB$ to $(A/_LC)/_R(B/_LC)$ is not an instance of **g**; that can be seen by the fact that **g** in turn is a unary version of function composition. Similarly, we cannot have a mapping of $A/_LB$ to $(A/_RC)/_L(B/_RC)$; this is the Geach equivalent to the other mixed composition.

While I believe there is evidence for breaking function composition down into the two steps of **g** plus application (e.g, see Sec. 5.1), most of the rest of this paper will continue to use function composition for purely expository reasons. Exactly the same points hold under the view that this is just a convenient way to collapse two steps, but the use of composition shortens the exposition. But the reader should keep in mind a when we say 'there is no mixed composition' that also means there is no Geach equivalent to this.

2.6. "Wh" (left) Extraction

In the CG and related literature (e.g., GPSG, HPSG, etc.) there has been a certain amount of debate as to whether the ordinary 'slash' of CG (indicating an argument slot, and here given directional features) is to also be used for the kinds of "leftward extractions" that go under the rubric of *wh* extraction (by which I include Topicalization, even though it contains no overt *wh*). Gazdar et al 1985 and earlier Gazdar 1979 clearly hinted at the similarity by naming the extraction feature that was used in GPSG as 'slash' (obviously not a coincidence). But those proposals were not embedded in a Categorial Grammar, and so the two were treated differently, and there was no claim that an argument slot 'missing' in a *wh*-type extraction construction had the same status as an expected argument in CG.

However, Steedman 1987 famously united them within CG by simply having many items select (to their right) something with a right slash. For example, a relative pronoun like *who* can have as its category $(N/_L N)/_R(S/_R NP)$ - it takes to its right an $S/_R NP$ to yield a N modifier, where the expression of category $S/_R NP$ is composed in exactly the same way that we find in, e.g., RNR cases. (Note incidentally that the R-slash on the $S/_R NP$ selected by *who* has be to listed on *who* in the lexicon as it is an exception to the general word order rules; the other slash directional features are the predictable ones. This particular fact itself does detract slightly from the appeal of collapsing 'extraction' slashes and the ordinary CG slash, but it is only a slight complication.) This proposal is certainly appealing; all other things being equal the Steedman approach is obviously simpler than having the argument structure slash and the 'extraction slash' be different. That said, all other things are not equal: it has often been noted that there are differences between RNR (which is a matter of conjoining two things with a right slash) and *wh* extraction (see, e.g., McCawley 1982, McCloskey 1986). One well known example is that extraction is possible in cases like *believe* __ *left,* but neither Heavy NP Shift nor RNR are allowed:

(14) a. the student who I believe was guilty of cheating

b. *I believe was guilty of cheating the student who sat in the front row.

c. *I believe was guilty of cheating and know should be reported to the dean the student who sat in the front row.

Steedman to appear does provide an interesting account of these differences while still taking the 'extraction' slash and 'expected argument' slash to be the same but that account requires several assumptions not compatible with the main results here (see fn. 8 for some discussion). Here I take these contrasts as support for the two 'slashes' being different; additional support for this is given in Sec. 4.

Hence I follow Oehrle 1990 in treating the two differently with a CG framework. I will use the notation | to indicate an 'extraction' gap; think of this as a feature (much like the superscript feature used for pronouns in Jacobson 1999 and other places, where A|B is an expression of category A 'missing' within it a B. Exactly the best way to treat the 'percolation' of information that something is missing in ordinary *wh*-type extraction cases is not our main focus, so we adopt some rough and ready first pass conventions. Let any A/B to map to A|B, and assume additionally that for every A/B there is also a variant (A|C)/(B|C). The mapping of A/B to A|B has no effect on the semantics (since both are of semantic type <b,a>, while the semantics of the second mapping above is the Geach rule. This alone is undoubtedly far too general: it allows wh-extraction gaps virtually anywhere.

It has no account of island effects, including not building in a Left Branch condition (although the examples below regarding "Lakoff chains" actually show that there are cases where | can go on a left branch), it provides no account of *that-trace* effects, etc. But we leave those matters here as this is not the main focus of the account, and for now will content ourselves with what is surely an overly permissive account of *wh* extraction.

3. Coordinate Structure Constraint, Lakoff chains, and Null coordination

Given the claim that *and* and *or* have the category (X/X)/X, there is no real reason to expect Coordinate Structure Constraint effects – something extra would be needed to block 'extraction' from a conjunct. After all, *and* is really no different from *think* in taking one argument to its right and another to its left and of course extraction is possible from at least the right argument of *think.* So - regardless of the ultimate account of extraction - without something extra there is no reason to expect the oddness of (15):

(15) *In which lake will Lee finish eating breakfast and swim?

Of course this remark holds for just about any theory of extraction, which is exactly why the Coordinate Structure constraint is generally stated as something extra, not something that falls out automatically from a theory of extraction in general.

But there is a large body of literature spanning decades speculating that CSC effects are about information structure rather than an actual syntactic violation, along with a large body of counterexamples to claim that the CSC is hardwired into the syntax (however that would be done). A typical and often cited one is (16), (17) is along the lines of examples from Goldsmith 1985 (who defends the CSC):

(16) What did Lee go to the store and buy __?

(17) How much beer can Lee drink __ and still stay sober?

Though less discussed, one can also construct counterexamples *or*:

(18) This is the car that Lee will (either) buy __ or else he'll just keep taking the bus.

(As to the placement of *either*, see Sec. 6.).

One reaction to these counterexamples is that we are dealing with a different *and* in these examples (and hence presumably also a different *or* in (18)) (see Goldsmith 1985 for a restructuring analysis). For a case like (16) various authors have said that this a 'subordinating' *and* rather than a coordinating *and*; presumably the hope would be to extend that to (17) and (18). But in the first place, it's not clear what it would mean for there to be a 'subordinating' *and* distinct from the normal *and* since under many modern views of the structure of *and* it is subordinating (any theory with only binary rules has that as a consequence).[9] Second, it's not clear what would be the meaning of this additional *and* that would cover all the cases. It sometimes seems to be compatible with meaning *and then* (so-called narrative' *and)*, but that can't be right for (17) where the two 'events' are contemporaneous. Note that these need not involve just VP conjunction; (19) - (21) are S conjunction even with different subjects, nor does there need to be a coreferential pronoun in the second conjunct as long as the connection between the two is clear:

(19) How many banks can Clyde rob __ and the police still not arrest him?

(20) How many banks can Clyde rob __ and Bonnie still pretend that all is fine?

(21) This is the castle that Charles will either have to decide to keep __ or William and Kate will just have to give up their dreams of being able to live there.

And what about the new *or* needed in (18) and (21)? It seems to have the same meaning as ordinary *or.* Again there is a lengthy literature on this (with respect to *and)* and I have nothing new to offer on just what is the needed information structure; see e.g. Kehler 1996, Altshuler and Truswell, 2022 among many others. Incidentally, none of these reduce to the phenonemon of 'coordination of unlikes'. In all of those standard cases, two expressions of different categories can be conjoined only in environments where each category itself is licensed. This is not the case for these CSC violations

What I take to be an especially strong argument for the existence of just a single *and* - as well as a single *or* - is that we find the same CSC-violating possibilities in examples with chains involving the null *and* and the null *or.* This was pointed out in Lakoff 1986 (for the *and* case; he did not look at *or*). His own use of these examples to provide an argument against a syntactic CSC was actually different than the point to be made here, but I nonetheless refer to these as ‘Lakoff chains’. By a Lakoff chain, I mean any series of at least 3 conjuncts/disjuncts where all but the lowest is null and where some contain gaps and some don't. Thus for the case of *and* he gives examples along the lines of those in (22) (I indicate which expressions have gaps and which don't by use of __, with no intended theoretical significance). (23) shows the same point with *or*:

(22) a. How much/What kind of beer did Lee go to the store, buy __, load __ onto his bike rack, and then bicycle home?

b. How much beer will Lee buy __, drink__ and still not fall asleep?

c. How much/What kind of beer did Lee buy __, bicycle home, and then instantly proceed to drink __?

(23) This is the car that Lee will (either) go ahead and buy __, decide he doesn't actually have enough money to spend on silly things, or convince his father to give him __.

I put *either* in (23) to make it more natural and easy to process, as it signals right away that the chain is an *or* rather than an *and* chain, but that is not strictly speaking necessary. (We return to *either* in Sec. 6.) The reason I take these to be strong evidence against the hypothesis that we happen to have additional lexical items (with a different syntax) pronounced the same as *and* and *or* is that it would be surprising to find that just these two (or more) new items also happen have null variants (also with this different syntax).

To elucidate the role of Lakoff chains in the material central to this paper, we first need an account of null *and* and null *or*. A simple account is to just have silent versions of these two. They will therefore have exactly the same meaning and word order possibility as overt *and* and *or* - just a different phonology, and I will write them as *~~and~~* and *~~or~~*. But there are two related complications to deal with. The first is that in a chain of these null items the lowest one must be overt. In other words, we get things like *I'm surprised to hear that Lee passed syntax, failed phonology, and dropped semantics* but not **I'm surprised to hear that Lee passed syntax, failed phonology, dropped semantics*. (When not embedded, one can sometimes get these chains without an overt *and* at the end but only in a certain style; these do not have the same distribution as the ones with the overt *and.* And there certainly is no variant without an overt *or* at the bottom.) Second, unlike the case of chains of conjunction with instances of overt *and* and overt *or* these cannot be mixed, as can be determined by the semantics. Thus *I ordered lobster, clams, and bananafish* only has the meaning with silent *and* and cannot be understood as 'lobster or clams and bananafish'.

This is true throughout the chain no matter how long– so there must be some way for the overt item (which occurs in the most deeply embedded position) to pass up the information that the semantics needs to be the *and* semantics all the way up (and similarly for *or*).

To this end, I adopt a feature passing solution which allows things that conjoin with either *and* or *or* to pass up the information needed for the semantics. Thus we adopt two features [&] and [v] and refine the categories for overt *and* and *or* such that the category for *and* is (X[&]/X)/X. (Recall the earlier discussion about features: if X itself is complex the [&] feature will actually be on the result.). *Or* is the same but with the [v] feature instead. This means that any ordinary conjoined expression (as in *Lee and Sandy*) has this feature on the category of the whole expression; I see no harm in that as long as it is understood that any item selecting for, e.g., an NP can take an NP[&] as well (and similarly for all other categories). This would hopefully follow from a more detailed theory of features and their role in the system, but we leave that here. Secs. 5.2 and 6 will show actual benefits to having these features on the 'top' category of conjoined/disjoined material. As to the role of these features on the case of null conjunction, we simply posit that the category of *~~and~~* is not exactly that of *and.* The full lexical entry for *~~and~~* (once directional slashes have been added) is: $\langle[\varnothing], (X[\&]/_L X)_R/X[\&]\rangle$. This ensures that *~~and~~* combines only with something as its right argument that has an overt *and* 'at the bottom', and it also passes up that feature just as does ordinary *and.* All of the same holds for *~~or~~*.

There is one problem to note here. For overt *and*, it was ensured that the innermost slash had a left feature on it because that was a modifier category - in other words *and semantics* (as in *phonology and semantics)* takes an NP to give an NP and so it follows that it takes that NP to the left. Similarly, we need to ensure that ~~*and*~~ *phonology and semantics* also takes its next argument to the left (as in *syntax,* ~~*and*~~ *phonology and semantics)*. But technically the category of *and semantics* as well as of ~~*and*~~ *phonology and semantics* is not a modifier since it these can combine with any ordinary X (without the [&] feature) to give something with that feature. In other words, the argument of this expression and the result are not exactly the same category. Hopefully this depends on an exact definition of 'modifier' (i.e., an exact specification of the directional slash rules) which allows for a bit of featural mismatch. We leave open here how best to do this, but it would not seem to be insurmountable. Moreover, there is an interesting point that will have consequences later. Notice that if we are assuming that features like this are actually encoded on the result there is an interesting prediction: if we conjoin two VPs as in *danced and sang* we want the resulting VP to have the [&] feature - this is needed as this can be the argument of ~~*and*~~. Hence its category is S[&]/NP. [10] But this means that (somewhat counterintuitively) the [&] feature is propagated up, so that an ordinary sentence like *Lee danced and sang* is S[&]. Note that the feature goes no higher; it stops propagating up once we have an expression of the category of the result. So, for example, since *that* is CP/S it can take an S[&] but nothing propogates the feature further; *that Lee danced and sang* is just CP.

The Lakoff chain examples above are unsurprising given the conventions above for the passing of the | feature. Notice that these chains provide evidence not only against a syntactic principle designed for Coordinate Structure Constraint effects,[11] but also against at least a completely general Left Branch Condition. Obviously some principle is needed for some such effects, but if it is too general it will incorrectly rule out some of the examples above. Even simple cases like (17 - 20) all involve a gap in the left conjunct.

Especially interesting for our purposes is to note that the Lakoff chains in these *wh*-extraction cases involve a complete what I will call 'mix and match'. In any two coordinated constituents, the gap can be on the left and not on the right (*load __ onto his bicycle and drove home* as in (22a)), on the right and not on the left (*bicycle home and then proceed to drink__* as in (22c)), or on both (as in ordinary ATB cases). Moreover, the lowest one can contain a gap or not, as is also true of the highest one. When we turn to right composition (the examples typical of RNR) and then to left composition (the "Maxwell/Manning" type examples) we see that these do not show the same freedom.

4. Rightward extraction, Lakoff chains, and "function composition" as function composition

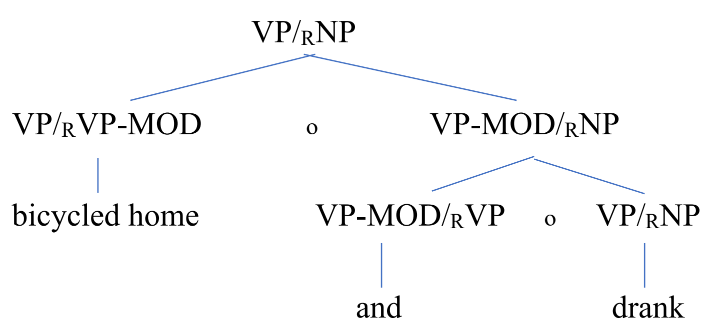
We now explore what happens in Lakoff chains in the case of rightward ‘extraction’ – i.e., cases of what is generally thought of as RNR, with more than just two conjuncts/disjuncts and where all but the bottom one involves silent ~~*and*~~ or ~~*or*~~. Here we find that ‘mix and match’ is also allowed in general - although we will see that it is for a different reason than the case of leftward *wh*-type extraction. But here an interesting constraint emerges: the rightmost member of the chain has to contain a gap. Before showing why that is true, let us first consider a good case like (24):

(24) Lee went to the store, bought __, bicycled home, and drank __ 2 sixpacks of beer.

The bottom of the chain in (24) is *bicycled home and drank* __, where we have a full VP on the right conjoining with one with a 'gap' - or, put differently, with the ordinary transitive verb *drank* which is of category $VP/_{R}NP$.

The expression *bicycled home and drank* can combine by function composition. For expository convenience, we use the abbreviation VP-MOD for $VP/_{L}VP$ (note that this is the category of, e.g., *and walks*). Note moreover that an ordinary VP like *bicycled home* can lift to $VP/_{R}(VP/_{L}VP)$ which we also abbreviate as $VP/_{R}$VP-MOD. In the illustration below of the derivation we begin with the lifted category of *bicycled home* to save space, and so the whole thing is analyzed as follows (we suppress the & feature):

(25)



Given the derivation above of *bicycled home and drank,* two things are possible as we compose bigger expressions. This expression can combine with a VP on its left in exactly the same way - using either overt *and* or, in the cases of interest here, ~~*and*~~. This for example is what happens in (26), where *bicycled home and drank* __ combines with *found his bicycle* in the same way as happens at the bottom of the chain as shown in (25).

(26) Lee bought __, found his bicycle, bicycled home, and drank __ 2 sixpacks of beer.

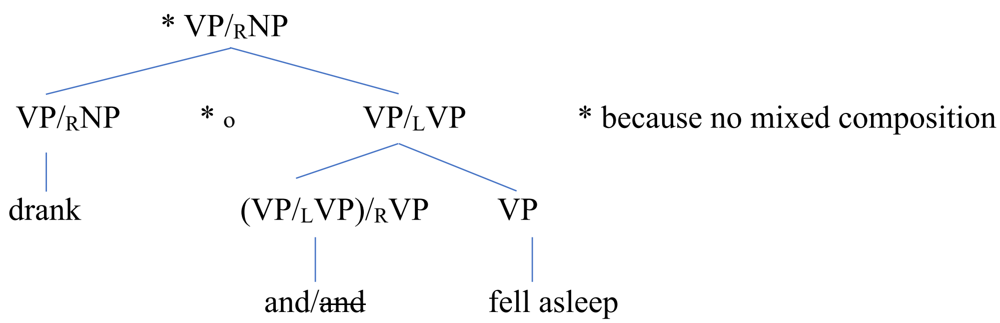
Alternatively, *bicycled home and drank* __ can combine with another $VP/_{R}NP$ by the ordinary coordination of likes; this is exemplified in (24) where *bicycled home and drank* conjoins (by ~~*and*~~) with *bought* __. Coordination of likes is also what happens in (26) when *found his bicycle, bicycled home, and drank* __ conjoins with *bought* __. So, once the chain is launched, we can keep mixing expressions with a gap with ones without.

We have dealt here only with combinations of $VP/_{R}NP$ (transitive verbs, whether simple or complex) and VP, but we should also find this with $S/_{R}NP$ and S. And we do:

(27) George loved __, Paul hated ___, Ringo was just like 'whatever', and John insisted on performing __ that silly song about a yellow submarine.

While it is interesting that these can all be handled in the CG account, there is so far no obvious advantage over treating them in the way they are sometimes accounted for in movement theories, which is by extraction to the right. Once such accounts have a treatment of ATB extraction and cases that don't show any "CSC effects", then these add nothing new to what we have already seen for the case of Lakoff chains with leftward *wh*-type extraction. But there is a striking prediction made by the account here which does not immediately follow (as far as I can see) in a rightward extraction account of RNR. The examples above show that a VP on the left can combine with a $VP/_{R}NP$ on the right (with *and* or ~~*and*~~ as intermediary) and the result is $VP/_{R}NP$. This is because the category of *and*/~~*and*~~ (both of which want a right argument first) can function compose with the $VP/_{R}NP$. ((27) shows the same thing with combining S with $S/_{R}NP$.) But there is no way for an 'ordinary' VP – i.e., a VP on the right to combine with a $VP/_{R}NP$ on the left – for the simple reason that this would not be real function composition - it would have to be mixed composition. So, for example, take *drank* which is of category $VP/_{R}NP$ and *fell asleep* of category VP. *and*/~~*and*~~ can of course take the VP on its right to give $VP/_{L}VP$. But there is no way for *and/~~and~~ fell asleep* to then take a VP/$_{R}$NP to its left without using mixed composition; my liberal use of * here is to indicate impossible combinations:

(28)



Notice that if there were mixed composition of the sort proposed in Steedman 1987 and to appear, the above is exactly what we would get: the VP/$_{L}$VP *and/and fell asleep* would take to its left the VP/$_{R}$NP *drank;* this would yield VP/$_{R}$NP by Steedman's use of the directional slash as a feature on the argument. But we stress that mixed composition is not true function composition and so is precluded under the view of categories advocated here.

Is this prediction correct? Indeed it is; compare (29b) to (29c) and (27) to (30):

(29) a. *Lee drank __ and fell asleep 2 sixpacks of beer.

b. *Lee went to the store, bought __, drank __ and fell asleep 2 sixpacks of beer.

c. Lee went to the store, bought __, bicycled home and drank __ 2 sixpacks of beer.

(30) *Paul loved __, George hated ___, John insisted on performing __ and Ringo was just like 'whatever' that silly song about a yellow submarine.

Thus while RNR cases do allow 'mix and match' in a Lakoff chain once it is launched, it has to be launched by having the 'smaller' item (the one of category A/$_{R}$B) being the rightmost one. This contrasts with the case of leftward *wh* type extraction. By way of a complete minimal pair, compare (22b) to (31):

(31) *Lee will buy__, drink__, and still not fall asleep 2 sixpacks of beer.

These contrasts also provide interesting new evidence that the two require different conventions (beyond various evidence already in the literature).

5. Leftward function composition chains

5.1. The predicted and actual existence of left looking chains

As noted above, a chain with a rightmost 'filler' is launched by something of the form *A and* $A/_{R}B$ (of course $A/_{R}B$ *and* $A/_{R}B$ is also possible). The following all exemplify some of the possible rightward looking Lakoff chains (followed by the B 'filler')

(32) a. [A ~~and~~ [$A/_{R}B$ and $A/_{R}B$]] B

b. [A ~~and~~ [A and $A/_{R}B$]] B

c. [$A/_{R}B$ ~~and~~ [A and $A/_{R}B$]] B

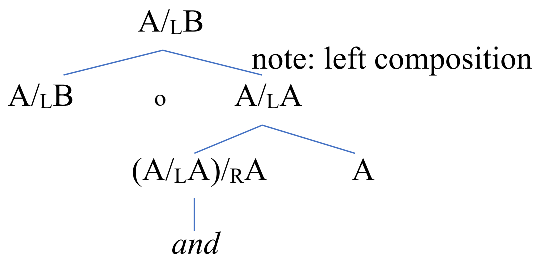
Consider the case in (32a). The mirror image - in terms simply of the string order - would be (33). (The bracketing is not mirror image because of the structure of *and* and ~~*and*~~[12]).

(33) B [$A/_{L}B$ ~~and~~ [$A/_{L}B$ and A]]

This raises two questions. Should we expect to find such cases? And do we find them? The answer to both is yes - and perhaps somewhat surprisingly so.

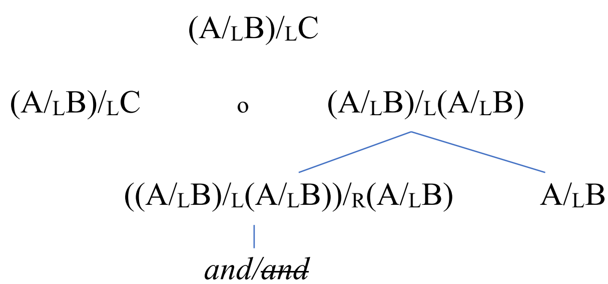
Turning first to the question of whether we should expect these, we can show that a left-looking Lakoff chain can be launched as shown below:

(34)



This can then conjoin with another $A/_L B$ by ordinary coordination of likes, and this can be either an overt *and* or an instance of ~~*and*~~; it is the latter that gives what I am calling a Lakoff chain. And notice also that an expression of any category $A/_L B$ itself could combine with something smaller to its left such as $(A/_L B)/_L C$ in exactly the same way:

(35):



We would thus expect to find chains not only of the form in (35) but even more complex ones of the general form in (36):

(36) B C [$(A/_L B)/_L C$ ~~and~~ [$(A/_L B)/_L C$ ~~and~~ [$A/_L B$ and A]]]

One can, of course, keep making these more complex, and add in instances of ordinary conjunction of likes within these chains. (One might wonder whether the added complexity to the left has to be incremental: might we also expect cases where, for example, an expression of a category $(A/_L B)/_L C$ can conjoin with an A to its left? Indeed such chains should (and do) also exist; we return below.)

Let us now turn to our second questions: do cases of the form illustrated in (33) and (36) actually exist? In fact, they do. As shown (briefly) in Maxwell and Manning 1996 cases like (33) exist, but have hitherto received little attention in the literature. A simplified version of Maxwell and Manning's case involves S/NP (i.e., VPs) and S:

(37) a. Lee went to the store, ate potato chips, and he danced.

NP [$S/_L NP$ [~~and~~ $S/_L NP$ [and S]]]

b. Lee went to the store, he ate potato chips, and he danced. = NP [VP, [VP and S]]

We can also construct cases with ordinary NP objects conjoining with full VPs; this is because an NP can be lifted to look for a transitive verb to its left; i.e. to category $VP/_L TV$:

(38) Barry plays the hammered dulcimer, the standup bass, and does magic tricks.

TV [$VP/_L TV$ [~~and~~ $VP/_L TV$ [and VP]]]

And we predict that there can be a chain of ordinary NP objects (again these are lifted) followed by a full VP, followed by a full S (again we return to NP objects conjoining directly with a full S to its right) - this is exactly an instantiation of (36):

(39) Barry plays the hammered dulcimer, the standup bass, does magic tricks and he builds treehouses.

NP TV [$(S/_L NP)/_L TV$ [~~and~~ [$(S/_L NP)/_L TV$ [~~and~~ $S/_L NP$ [and S]]]]]

As shown by the 'glosses' below each example, each involves an instantiation of the general schema shown above. For example, take the expression *ate potato chips and he danced* in (37a). *and* is the item of category $(S/_L S)/_R S$ and so it combines with *he danced* to give $S/_L S$. *ate potato chips* is of category $S/_L NP$ so this can combine with *and he danced* by left composition. This gives the coordinated expression *ate potato chips and he danced* which itself is an $S/_L NP$ (i.e., an ordinary VP). This then combines (in the two steps expected by the binary ~~*and*~~ by ordinary conjunction of likes, to give the VP *went to the store, ate potato chips, and he danced.*

The case in (38) is a bit more complex but only because it involves lifting the object NP. Here, *the hammered dulcimer* can conjoin with VP (giving *the hammered dulcimer and does magic tricks)* as follows. *the hammered dulcimer* lifts over transitive verbs; its lifted category is $\mathrm{VP}/_{\mathrm{L}}\mathrm{TV}$. *and does magic tricks* is the ordinary "VP-modifier" - i.e., $\mathrm{VP}/_{\mathrm{L}}\mathrm{VP}$. From here we just have another instance of left composition: *the hammered dulcimer and does magic tricks* wants a transitive verb to its left to give a VP (hence of category $\mathrm{VP}/_{\mathrm{L}}(\mathrm{VP}/_{\mathrm{R}}\mathrm{NP})$). This then combines with lifted *the standup bass* by normal coordination of likes. Hence the expression *the standup bass, the hammered dulcimer, and does magic tricks* wants a transitive verb to its left. It finds *plays* to yield an ordinary VP.

The question was raised earlier as to whether the 'growth' needs to be incremental: do we, for example, have a chain of the form *(lifted) NP, (lifted) NP, and S*? The answer is yes: such an example is (40) (the use of a different subject for the final S is to avoid any potential complexities engendered by having a pronominal subject):

(40) Barry plays the hammered dulcimer, the standup bass, and Patty plays the flute.

NP TV [$(\mathrm{S}/_{\mathrm{L}}\mathrm{NP})/_{\mathrm{L}}\mathrm{TV}$ [~~and~~ [$(\mathrm{S}/_{\mathrm{L}}\mathrm{NP})/_{\mathrm{L}}\mathrm{TV}$ and S]]].

The analysis of this under the parse shown above is not immediate using just function composition, but presents no problem if we instead recast function composition as Geach + application. Recall that *and S* can compose up to be $\mathrm{S}/_{\mathrm{L}}\mathrm{S}$. We need then only two applications of "Geach". The first maps this to $(\mathrm{S}/_{\mathrm{L}}\mathrm{NP})/_{\mathrm{L}}(\mathrm{S}/_{\mathrm{L}}\mathrm{NP})$ (a VP modifier). The second maps that to $((S/_LNP)/_LTV)/_L((S/_LNP)/_LTV)$. Hence *and Patty plays the flute* takes *the standup base* as argument. From there we have coordination of likes.

If we were to insist on using function composition rather than the two-step Geach + application, we still can get the sentence (40) in a different way. This can be put together by having *the hammered dulcimer* raise not over TVs but over $S/_RNP$ - i.e. it lifts to category $S/_L(S/_RNP)$. And, as shown above, *and Patty plays the flute* is $S/_LS$. These two left compose exactly parallel to the previous example, giving *the hammered dulcimer and he does magic tricks* as a well-formed expression of category $S/_L(S/_RNP)$. *the standup bass* can lift in the same way and we have coordination of likes in the expression *the standup bass, the hammered dulcimer and Patty plays the flute.* So this too wants an $S/_RNP$ to its left to give an S. How does it find such an expression? Recall that *Barry plays* can compose up (with lifted *Barry*) to give an expression of exactly this category, and that then becomes the left argument of the Lakoff chain.

It is worth noting that the actual example in Maxwell and Manning 1996 involved not just a simple object but a "fancy" one of the type analyzed in Dowty 1988:

(41) John wanted to study medicine when 11, law when 13, and to study nothing at all when 18.

(Incidentally, their analysis of these has virtually nothing in common with the analysis here nor did they discuss the phenomenon in any generality.) The analysis of (41) is basically the same as the analysis of (38); *medicine when 11* can compose in the way shown earlier in (38) to be something wanting a TV to its left to give a VP, so the rest is the same as the case of an ordinary object which lifts over TVs. And unsurprisingly - parallel to the "Barry" sentence in (39) except with 'fancy' objects - we find chains like (42):

(42) Laura teaches syntax in the fall, compositional semantics in the spring, goes to music workshops in the summer, and she spends each sabbatical writing a book.

These examples illustrate two points of note. First, it is clear that these chains need not involve 'narrative *and*'; they do not require an "and then" reading (although in all cases there is of course a pragmatic pressure to sequence the conjuncts chronologically where possible). Second, (as already shown in (40)) when there is a full S at the end it can have a different subject from the original one. Both points are illustrated in (43) where the temporal sequence is not one of time moving forward (indeed, between most of the conjuncts there is no temporal sequence), and where the last S has a different subject.

(43) A: Boy your friends Patty and Barry are really talented!

B: They sure are. Barry built a standup bass during the pandemic, a hammered dulcimers in the 90s, does great magic tricks, and Patty plays all sorts of wind instruments including even a crumhorn.

` Because the syntax and the semantics track each other and involve parallel combinatorics, the semantics or all of these examples works out exactly as expected. Let me illustrate (very briefly) with just one case: *plays the hammered dulcimer, the standup bass, and does magic tricks.* Given the generalized semantics for *and* in Partee and Rooth (1983), [[and does magic tricks]] is $\lambda P_{<e,t>}$[P ∏ does magic tricks] - of type <et,et>. But this combines not with an <et> meaning but with something of type <eet,et> via function composition, yielding $\lambda R_{<eet>}$[λR[R(the hammered dulcimer) ∏ does magic tricks]. The same thing happens further up. Leaving out each step in order to save space, the meaning of *the hammered dulcimer, the standup bass, and does magic tricks* is $\lambda R_{<eet>}$ [R(the standup bass) ∏ R(the hammered dulcimer)] ∏ does magic tricks. [[plays]] eventually fills in the R slot, and we get the intersection of playing stand up bass, playing dulcimer and doing magic tricks, exactly as we want.

5.2. The "Grow Only Rightward" phenomenon: No Mixed Composition

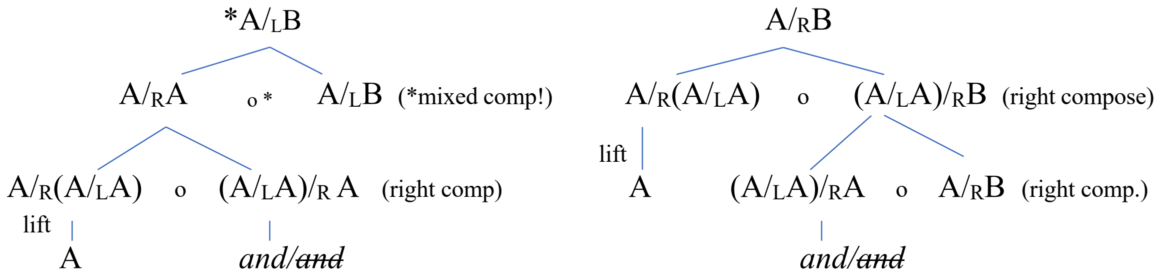
Recall that in the case of a Lakoff chain looking for something to its right, the rightmost one had to be the smallest, but other than that any kind of mix and match is possible for reasons discussed above. The left looking case, however, is different. We have seen how larger expressions on the right (lower down) can combine with smaller ones to their left; that is the cases above are all 'launched' by the bottom of a chain being an expression of the form $A/_L B$ *and* A where that expression itself is $A/_L B$. (As discussed above - chain can also be launched by, e.g., as $(A/_L B)/_L C$ *and* A and presumably even 'smaller' things as left conjunct.) But there is an interesting prediction: such an expression should not be able to combine with something larger to the left (by "larger" we mean a category like A as opposed to A/B). The difference between the right and the left situation is ultimately due to the asymmetry *and* - which asks to take its first argument to the right and then the next to the left - combined (crucially) with the lack of mixed composition.

To exposit, let us try to compose an expression of the form A *and*/~~*and*~~ $A/_L B$ - either at the beginning (rightmost, lowest) part of the chain (which would involve overt *and*) or anywhere higher up (using ~~*and*~~). (If this can't be done, it follows that we also can't get something like A *and*/~~*and*~~ $(A/_L B)_L C)$). Consider first what it would take for *and* (or ~~*and*~~)

to combine on the right with an expression of category $A/_L B$. First, we could use the version of *and/~~and~~* of cagetgory $((A_L B)/_L(A/_L B)/_R(A/_L B)$. This is fine but if nothing else happens, this just involves coordination of likes. After we have the expression *and x* we have an $A/_L A$ modifier - i.e. something waiting for that on the left, Obviously we cannot function compose *and* $A/_L B$ with *A*, since the latter isn't a function. But we can always raise the expression of category A to either of the following functions: $A/_L(A/_R A)$ or $A/_R(A/_L A)$. But that is of no help. An expression of the form *and* $A/_L B$ is category $(A_L B)/_L(A/_L B)$ and has no way to combine with one of those lifted categories. Can we instead start with *and* of category $(A/_L A)/_R A$ and function compose this with $A/_L B$? The answer is no, that too would involve mixed composition.

What about trying a different tack, which is really as close as possible to the mirror image of what happens in the right looking case. The way we can try to simulate a mirror image is to have *and* combine with its arguments in a left branching fashion. After all that should be allowed by function composition just the same way that we have a left branching derivation of, e.g., *Lee tells lies* (as discussed in Sec. 2.5.2). The problem here is that the next step once again would require mixed composition. We can illustrate this, comparing the right looking derivation, in (44b) with the left looking one in (44a). The categories shown in bold are the ones we are ultimately trying to conjoin as well as the category of the hoped for result, and the label at the top of the whole left is of course illegitimate (hence indicated with a * prefix), as it requires the illegal mixed composition:

(44) a. Left looking b. Right looking



Hence if there is no mixed composition, this derivation doesn't exist. While there are other possible things that one might try, there is no way to get a larger expression to the left of a smaller one, not only at the beginning of the chain, but anwhere higher up. It is thus precited that the only way to get leftward Lakoff chains is as shown in Sec.5.1. Things can grow rightward, but they cannot shrink.

Is this prediction correct? Strikingly, it is. For example, consider (45b) where an object or a 'fancy' object follows a VP in the chain:

(45) a. Sarah teaches Lexical Semantics in the fall, Compositional Semantics in the spring, and plays the violin in her spare time.

b. *Sarah teaches Lexical Semantics the fall, plays the violin in her spare time, and Compositional Semantics in the spring.

If (b) is good at all, it is only on the pragmatically bizarre reading in which Lee plays Compositional Semantics (we return to that below). Similarly (46b) - if good at all - can only be understood on the strange reading where Barry plays treehouses.

(46) a. Barry builds dulcimers, treehouses, and plays the standup bass.

b. *Barry builds dulcimers, plays the standup bass, and treehouses.

The same point should hold in the case of chains of VPs and full Ss. At first glance, it appears however that this generalization does not hold in that case; (47b) is better than expected (I have consulted several speakers; most find it pretty good although not perfect), and this might appear to threaten the above prediction:

(47) a. Sally teaches semantics, studies astronomy, and she plays the crumhorn.

b. ? Sally teaches semantics, she plays the crumhorn, and studies astronomy.

In fact, though, one can show that this involves a different and somewhat surprising analysis - the same analysis which allows for the pragmatically bizarre readings of (45b) and (46b). My claim is that *plays the crumhorn* and *studies astronomy* are conjoined VPs which then take *she* as subject to give an S. In other words, that this has the structure (48a) rather than the structure (48b) which necessitates mixed composition:

(48) a. Sally teaches semantics ~~and~~ [$_S$ she [$_{VP}$ plays the crumhorn and studies astronomy]].

b. Sally teaches semantics ~~and~~ [[$_S$ she plays the crumhorn] and [$_{S/LNP}$ studies astronomy].

I have argued that (48b) is impossible as it involves mixed composition. But one might expect (48a) to also be impossible as one's first intuition is that the conjoined VPs should not allow the [&] feature to keep percolating up; the domain of the feature should stop at the VP level. That the highest VP itself has the feature [&] is predicted by the feature passing conventions adopted earlier. It needs to be on that VP since that VP can of course conjoin with another VP by ~~*and*~~*,* but one's first intuition is probably that things should stop there; it should not be able to percolate up to the S *she plays the crumhorn and studies astronomy.* Or should it?

Recall the discussion in Sec. 3 when we introduced the needed feature analysis of *and* and *or* in order to license and give the correct semantics for *~~and~~* and *~~or~~*. The need for some such feature analysis is probably theory-neutral (unless one has an entirely different analysis of conjunction/disjunction in general). But there is a piece of the analysis discussed there which is unique to CG: the speculation that whenever we have features on complex categories, they are encoded on the result. Incidentally, this speculation was not motivated by the material here: it is the cleanest way to think of features in CG as it gives a clean account of how they are 'passed up', and follows if the 'basic categories' themselves are feature bundles.. If that is correct, though, it makes a striking prediction: the S *she [plays the crumhorn and studies astronomy]* should actually have the feature [&] because *plays the crumhorn and studies astronomy* is of category S[&]/NP. When it takes the NP subject the feature remains on the S result.

And in fact we can show that the structure in (48a) (conjoined VPs) rather than the illicit mixed composition structure in (48b) is indeed the correct analysis of (47b). We need merely consider a case with different subjects:

(49) a. Barry builds dulcimers, does magic tricks, and Patty plays the flute.

b. Barry builds dulcimers, Patty plays the flute, and does magic tricks.

(49) only means that Patty does magic tricks - exactly the expected semantics given the structure in (48a). It cannot mean the same thing as (47a), so cannot just be a case of the chain being launched by *S and* $S/_{L}NP$ which would then conjoin with a higher S using *~~and~~*. The same remarks hold for the VP cases.

I am a bit reluctant to a total victory here for CG (combined with the assumption that features are always encoded on the result categories - a principle which is meaningful only in CG and which, if correct, would give yet further evidence for the CG view of categories). The reason is because I and others don't find sentences like (47b) to be perfect - as indicated by the ? notation there. Several others that I have informally consulted had the same feeling - that these are surprisingly good but not perfect. The analysis given above - which depends on a particular assumption about feature passing - predicts that these should be perfect. So we leave this in need of further discussion (and perhaps much more informant work with naive informants - it could well be that the expectation of a linguist is driving the judgment of this not being perfect). It is clear, however, that the main point is secure: (47b) does not have the analysis in (48a) (as shown by the semantics of (49b). Hence the 'grow only rightward' prediction is supported by the empirical data. This prediction hinges on the unavailability of mixed composition and that, in turn, follows from the view promoted here that syntactic categories correspond to functions on strings, whereby mixed composition simply is not function composition.

6. *Either*

Alan Munn (personal communication) points out to me that the material here is reminiscent of facts about the placement of *either* discussed in, among others, Larson, 1985, Munn 1993, Schwarz 1999, and den Dikken 2006. That is, *either* can be in a position which appears to be 'too low' as in (50) (call that "too low either") or in a position which is too high as in (51) (call that "too high either"):

(50) a. Barry will play either the hammered dulcimer or (he will) do magic tricks.

b. Barry will play either the djembe or Patty will use the foot tambourine.

(51) a. Barry will either play the hammered dulcimer or the standup bass.

b. Either Barry will play the hammered dulcimer or the standup bass.

Den Dikken (unlike Larson and Schwarz) gives a unified analysis of the 'too low' and the 'too high' case; space precludes a serious discussion of his analysis which is quite different from the one I will propose here. Whether or not my analysis unifies the two cases depends on a fuller analysis of 'too high' *either*; the remarks here are intended as preliminary. But strikingly, the existence of too low *either* follows (almost) immediately from the apparatus in this paper, all of which was developed completely independently of this domain.

We need only one assumption not already made but one which is obvious: that *either* selects for an expression with the [v] feature, so let it be X/X[v] (we return to its meaning). Since we need the feature [v] anyway to account for the distribution of silent ~~or~~, it is hard to imagine any simpler account of *either* than just to have it take advantage of the fact that expressions with an *or* lower down will wear that fact on their sleeves. Incidentally, the use of the [v] feature in the distribution of silent ~~or~~ and the use of this in the distribution of *either* are completely theory-neutral proposals; Categorial Grammar plays no central role in either of these bits of apparatus. The analysis of 'too low' *either*, on the other hand, does rely on the CG apparatus explored here.

Hence, with the above rather theory-neutral assumption about feature passing and the category for *either* - combined with the basic function composition (or Geach) apparatus explored throughout this paper - the existence of 'too low' *either* is automatic.

Consider (50a). We already know that *the hammered dulcimer or do magic tricks* (*lifted NP or VP*) can compose up to give (S/$_L$NP)/TV <[v]>. We know this independently that this can combine with ~~or~~ to give expressions like *the standup bass, the hammered dulcimer, or do magic tricks.* So it is perfectly possible instead for the expression *the hammered dulcimer or do magic tricks* to combine with *either* as in (50a). The cases in (50) where we have a full S are handled just like the Lakoff chain example in (40).

Not surprisingly 'too low' *either* also participates in Lakoff chains. Some speakers don't like *either* with more than two disjuncts as in (52), but many speakers (including myself) are fine with that; ((52) is a case of *either* being neither too high nor too low, nor is it strictly speaking a "Lakoff chain" as the disjuncts are all the same category):

(52) At the memorial service, either Lee will play cello, Sandy will read a poem, or Leslie will give a tribute.

The Lakoff chains in (53) are thus perfectly unremarkable:

(53) a. Barry will play either the dulcimer, the bass, or will do magic tricks.

b. Barry will play either the dulcimer, do magic tricks, or Patty will play flute.

Two further points to notice about *either.* First, as indicated by the category hypothesized above, it stops the passing of the [v] feature. We can show this even with the case of a string or disjunction of likes, but we need a three way chain to show this:

(54) a. *Barry will play the dulcimer, either the bass, or the flute.

b. *Barry will play the dulcimer, either do magic tricks, or build a treehouse.

Second, as a first pass we take its meaning as the identity function (but see below). It does for some reason reinforce the exclusive implicature of *or,* but (pace den Dikken, 2006) its meaning cannot be XOR. This is shown by the fact that - like ordinary *or* - the exclusive interpretation can be cancelled and goes away in downward entailing environments:

(55) a. He ate either the hummus or the guacamole - perhaps even both.

b. Everyone who ate either the hummus or the guacamole got sick.

What about too high *either?* Some such instances are also automatic here, but there is a rich literature both on its fine-grained syntactic distribution and interesting semantic effects that are beyond the current scope. Suffice it to say here that - given a tentative assumption we have made about features in CG - the case of (51b) is unsuprising, as is lower occurrences of *either* as in (56):

(56) Barry will either play the hammered dulcimer or the standup bass.

Looking at (51b), we see that if it is correct that features in CG are always encoded on the result, we have seen that *Barry will play the hammered dulcimer or the standup bass* will have the [v] feature on the highest S node. This is because each of the NPs must be lifted (for the *or* semantics to make sense), and so can each be lifted to $VP/_L TP$. When *or* combines with the lowest of these it propagates up the information that there is an *or* so that *or the standup bass* is $VP/_L VP$ with the [v] feature. But we have speculated that that always means that feature is on the final result, so what we have here is $(S[v]/_L NP)/_L(S/_L NP)$. This in turn means that *play the hammered dulcimer or the standup bass* is $S[v]/_L NP$ and *Barry (will) play ...* is S[V]. *Either* is thus perfectly happy to take this as argument. But we predict that 'too high' *either* cannot go further up, since that feature will not propagate above S, and thus, as expected, we do not find cases like (57):

(57) a. *Patty either/Either Patty thinks that Barry will play the dulcimer or the bass.

b. *Patty thinks either that Barry will play the dulcimer or the bass.

(Compare (57b to the same sentence with *either* under *that.).* Thus - with den Dikken (2006) - the prediction is that *either* cannot (to use the movement metaphor) 'cross' the boundary of a (tensed) S; it cannot go up to the CP level. (I assume that embedded infinitive VPs, as in *I either want to play guitar or recorder* is indeed an VP and not an S with pro subject - a theory defended as early as in, e.g, Dowty 1985, and so there is no actual S in this which would stop the propagation of the [v] feature.)

But the analysis is not complete. The reason is that while the account here predicts the existence of (51) and (56) (as opposed to (57)), the literature contains discussion of a number of constraints on 'too high' *either* which we have said nothing about. For example, Larson (1985) notes important semantic effects of 'too high' *either.* Thus consider the contrast between (58a) and (58b); (58b) only allows the reading in which *or* has widest scope - that is, the reading that can felicitously be followed by "but I don't know which" (there may be a wide scope de reading but the main point here is that on the de dicto reading, wide and narrow scope are both possible in (58a) but only wide scope in (58b):

(58) a. The department is looking for either a semanticist or a phonologist.

b. Either the department is looking for a semanticist or a phonologist.

This fact requires a more thorough understanding of the meaning of *either;* our earlier 'first pass' assumption that it is just the identity function is probably incorrect since its placement matters to the interpretation.

That said, we stress again that the existence of 'too low' *either* is automatic - these follow in the same way that left-looking Lakoff chains do, and at least a first pass at too high *either* is plausible here. But to summarize the more central point in this paper: the fact that the left-looking Lakoff chains can only 'grow rightward' is an automatic consequence of the general apparatus of CG assumed here, along with the claim that there is no mixed composition. And that latter fact is not just a stipulation - but a deep consequence of the view of syntactic categories taken here: they literally correspond to functions in <string, <string,string>>. All of this in turn provides interesting support for the general CG worldview.

Endnotes

[1] In accordance with reasonably standard terminology in CG (and many other places) I use the term NP; the reader can substitute DP if preferred.

[2] I am very grateful to Dick Oerhle for pointing this out to me, and for pointing out that with this view there is no mystery as to why, for example, lift rules preserve the order that they do, as will be discussed in Sec. 2.5.1.

[3] Steedman to appear also takes categories as functions, but not on strings: rather it is a function from categories to categories. This alone does not provide the results here.

[4] Actually in all cases there are an infinite number of derivations yielding the same meaning, since the subject and the VP can keep lifting over each other. I see nothing problematic about this since they all yield the same meaning.

---

[5]It has often been argued RNR is much more general and is not just about coordination - a particularly thorough discussion is in Chaves 2007. For example, we find cases like *The student who loves __ was assigned to mentor the student who hates __ OT phonology.* We have nothing to say about these cases here and hope that they can be handled in the same way that *wh*-type extraction parasitic gaps are. For example, Steedman 1987 provides an account of parasitic gaps using an additional combinatory principle within the general CG apparatus, and even though I am treating (leftwards) *wh*-type extraction differently from the rightward cases the same basic idea could be extended to the right cases. That said, I do not see at this point how to fold that operation into the general view of categories taken here, so I leave that open at this point. But even if it turns out that coordination *per se* is irrelevant to "RNR" cases, this has no bearing on the main points in Sec. 5.

---

[6] The reader may wonder about fancier cases like (i)

i. Lee cooked and Sandy ate scallops on Monday and lobster on Tuesday.

This can be put together by having *cook* and *eat* first lift from TVs to category $VP/_R(VP/_LTV)$. (Recall that TV is an abbreviation for $VP/_RNP$). As seen earlier, *Lee* and *Sandy* can both lift to the ordinary category for generalized quantifier subjects: $S/_RVP$. Since those can compose with *cook* (and *eat)* by right composition we get *Lee cooked* of category $S/_R(VP/_LTV)$, and similarly for *Sandy ate.* These two can cojoin, and *Lee cooked and Sandy ate* is of course also of category $S/_R(VP/_LTV)$. Note that $VP/_LTV$ is exactly the category of both a lifted object and also of a 'fancy object' like *scallops in Monday,* as well as being the category of the conjoined fancy object *scallops on Monday and lobster*

---

on Tuesday. It is therefore the right sort of thing to be argument of *Lee cooked and Sandy ate.* The semantics comes out as expected as an interested reader can verify.

Additionally, it is reasonable to wonder about cases like *I gave a backpack to Lee and a suitcase to Sandy.* While Dowty 1988 shows that these are straightforward if *give* first combines with the NP object and then the PP, we are assuming that most likely the NP object is introduced as an infix. We would thus need the system to work out in such a way that *a backpack to Lee* can compose up to take (most likely as a circumfix) a $TV/_I NP$ (so its category would be $TV/_C(TV/_I NP)$. It would take *give* as an 'infix' but presumably that would be vacuous in that its own infixation point would be on the left edge, so it would end up just taking *give* to the right. I suspect (and hope) that this would all work out with a full treatment of infixation/circumfixation including a formalization of the infix points.

[7] Actually even this is an oversimplification: not any category can instantiate X, as there are examples of strings which most theories would agree are constituents but which cannot coordinate. As discussed in Sex. 2.4, many theories including the one here treat *and Lee* as a constituent in *I recommended Sally and Lee.* Yet we do not get **I recommended Sally and and Lee and Sandy.* Similarly for: **I recommended Sally or and Lee and Sandy.*

[8] Steedman to appear does allow for free mixed (or, in his terms, 'crossing') composition giving a $VP/_L NP$ *believes left.* The reason this cannot combine with an NP to its left is because he assumes plain NPs never are introduced into a syntactic derivation: all NPs are type raised via a case marker (and, crucially also there is no free type raising). There is a marker (ACC) which maps an NP to $VP/_L(VP/_R NP)$ (for object NPs) but no marker yielding a category $VP/_R(VP/_L NP)$. Since these higher type categories (for subject NPs, object NPs)

etc. are not derived by Lift but by the lexical specification of the case markers, it is not clear why they should give categories that look just like those that Lift gives by definition. Moreover, as Steedman notes he needs a second ACC marker of category $S/_L(S/_R NP)$ to account for RNR cases like *Lee loves and Sandy hates model theoretic semantics,* somewhat undermining the beauty of having RNR follow immediately from function composition. (This marker also outputs a category that we would instead get by definition from a lift.) In any case, Sec. 5 presents new evidence against mixed composition.

[9] This problem holds also for Goldsmith's restructuring analysis since his 'restructured' structure is essentially the same as a modern view of the structure of ordinary conjunction.

[10] There is a second technical problem. Consider *After college, Lee will go to graduate school, join a fancy company, or [inherit a batch of money and travel the world].* The lowest conjoined VP has the category [S[&]/NP]. Indeed it needs that [&] feature since it could combine to its left with ~~*and*~~ and it will carry allong the 'and' semantic. But here it happens to find an overt *or* to its left, whose category is (X[v]/X)/X. And we certainly want *join a fancy company or [inherit a batch of money and travel the world]* to have the [v] feature on the final result to give the full semantics for the whole S. (Of course I have built in the meaning by showing overt bracketing; the string itself could have a meaning where *join a fancy company or inherit a batch of money* is a constituent and *and* has scope over all of it. But the point is that the meaning shown by the bracketing above is one possiblity.) The issue, then, is that there are two competing principles on the final result feature for *join..or [inherit..and travel]].* If *or* is the boss it should be the feature [v], but by inheritance from the lowest *and* it should be &. We will assume conventions by which

the two are mutually exclusive, and whereby when the feature passing conventions conflict with the lexical specification of, e.g., *or,* the latter takes priority.

[11] As is well known, there appears to be a strong principle which has the effect of ruling out 'extraction' *of* a conjunct (rather than just *from* a conjunct), and this effect does not follow immediately from most accounts of extraction (including the one here)

(i) a. *Which table did Lee build a matching chair and __?

b. *Which table did Lee build __ and a matching chair?

[12] One can get structures where *and/~~and~~* combines first with the left argument and then the right if the first combination there involves function composition. Unsurprisingly, the generalization to be discussed in Sec. 5.2 is not changed by this possibility; due to space limitations we simply invite the interested reader to verify this for themselves.

References

Altshuler, D. and R. Truswell 2022. *Coordination and the Syntax-Discourse Interface.* Oxford: Oxford University Press.

Bach, E. 1979. "Control in Montague Grammar", *Linguistic Inquiry* 10.4, 515-31.

Bach, E. 1980. "In Defense of Passive", *Linguistics and Philosophy* 3, 297-341.

Chaves, Rui P. 2007. *Coordinate Structures - Constraint-Based Syntax-Semantics Processing.* Ph.D. Dissertation, University of Lisbon.

Chomsky, N. 1957. *Syntactic Structures.* The Hague: Mouton.

Chomsky, N. 1970. "Remarks on Nominalization", in R. Jacobs and P. Rosenbaum (eds.), *Readings in English Transformational Grammar.* Waltham: Ginn, 184-221.

Den Dikken, M. 2006. *Either*-float and the syntax of co-ordination. *Natural Language and Linguistic Theory* 24(3):689–749.

Dowty, D. 1982. "Grammatical Relations and Montague Grammar", in P. Jacobson and G.K. Pullum (eds.), *The Nature of Syntactic Representation.* Dordrecht: Kluwer, 79-130.

Dowty, D. 1985. "On Recent Analyses of the Semantics of Control", *Linguistics and Philosophy* 8.3. 291-331.

Dowty, D. 1988. "Type Raising, Functional Composition, and Non Constituent Conjunction", in R. Oehrle, E. Bach and D. Wheeler (eds.), *Categorial Grammars and Natural Language Structures.* Dordrecht: Kluwer, 153-197.

Gazdar, G. 1979. "English as a Context Free Language". ms., Sussex: University of Sussex.

Gazdar, G., E. Klein, G.K. Pullum and I. Sag, 1985. *Generalized Phrase Structure Grammar.* Oxford: Blackwell and Cambridge, MA: Harvard University Press.

Goldsmith, J. 1985. "A Principled Exception to the Coordinate Structure Constraint", *Proceedings of the 21st Meeting of the Chicago Linguistic Society.* Chicago: Chicago Linguistics Society, 133-143.

Jacobson, P. 1987. "Phrase Structure, Grammatical Relations, and Discontinuous Constituency", in G. Huck and A. Ojeda (eds.), *Discontinuous Constituency* (Syntax and Semantics 20). New York: Academic Press, 27-69.

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

Jacobson, P. 2014. *Compositional Semantics: An Introduction to the Syntax/Semantics Interface.* Oxford: Oxford University Press.

Hendriks, H. 1993. *Studied Flexibility: Categories and Types in Syntax and Semantics.* Ph.D. Dissertation, University of Amsterdam Institute for Logic, Language and Computation.

Kehler, A. 1996. "Coherence and the Coordinate Structure Constraint" *Proceedings of the Twenty-Second Annual Meeting of the Berkeley Linguistics Society: General Session and Parasession on The Role of Learnability in Grammatical Theory*. Linguistic Society of America *eLanguage,* 220-231.

Kuno, S. 1987. *Functional Syntax - Anaphora, Discourse and Empathy.* Chicago: University of Chicago Press.

Lakoff, G. 1986. "Frame Semantic Control of the Coordinate Structure Constraint". Proceedings of the 22nd Annual Meeting of the Chicago Linguistics Society. Chicago: Chicago Linguistics Society. 152-167.

Larson, R. K. 1985. "On the syntax of disjunction-scope". *Natural Language & Linguistic Theory*, 3: 217–264

Larson, R. 1988. "On the Double Object Construction", *Linguistic Inquiry* 19:3, 335-391.

Maxwell, John T. and Manning, Christopher D. 1996. A Theory of Non-constituent Coordination based on Finite State Rules. In Proceedings of the First LFG Conference. Grenoble: CSLI Publications.

McCawley, J. 1982. "Parentheticals and Discontinuous Constituent Structure", *Linguistic Inquiry* 13. 91-106.

McCloskey, J. 1986. "Right Node Raising and Preposition Stranding", *Linguistic Inquiry* 17.1, 183-186.

Miller, P., 1990. "Pseudo-gapping and do so substition". *Proceedings of the Twenty-Sixth Chicago Linguistics Society* 1. Chicago: Chicago Linguistic Society, 293-305.

Montague, R. 1973. "The Proper Treatment of Quantification in Ordinary English", in P. Suppes, J. Moravcsik, and J. Hintikka (eds.), *Approaches to Natural Language.* Dordrecht 221-242.

Munn, A. 1993. *Topics in the Syntax and Semantics of Coordinate Structures.* Ph.D. Dissertation, University of Maryland.

Oehrle, Richard, 1990. "Categorial Frameworks, Coordination, and Extraction" in *Proceedings of the Ninth West Coast Conference on Formal Linguistics.* Stanford: Stanford Linguistics Department. 411-426.

Partee, B. and M. Rooth 1983. "Generalized Conjunction and Type Ambiguity", in R. Bauerle, C. Schwarze, and A. von Stechow (eds.), *Meaning, Use and Interpretation of Language,* Berlin: De Gruyter, 361-383.

Pollard, C. 1985. *Generalized Phrase Structure Grammar, Head Grammars, and Natural Language.* Ph.D. Dissertation, Stanford University.

Schwarz, B. 1999. On the syntax of *either ... or*. *Natural Language & Linguistic Theory*, 17:339–370.

Steedman, M. 1987. "Combinatory Grammars and Parasitic Gaps", *Natural Language and Linguistic Theory* 5, 403-439.

Steedman, M. to appear. "On Internal Merge", *Linguistic Inquiry.*