QUANTIFIER SCOPE AMBIGUITY

By: Aksel Ozer

I will first delve into the approach furthered by Heim and Kratzer in regards to scope ambiguity and quantifier movement. Before doing so, I want to look at the merits of a type shifting approach briefly, as the literature is replete with partial to full type shifting approaches. A type shifting solution to the problem of in situ quantifiers presented in Heim and Kratzer begins by modifying the denotations of the common quantifiers such as *everybody* and *somebody*. They create two different denotations for both *everybody* and *somebody* depending on whether they combine with one place or two place predicates. For one place predicates, *everybody* and *somebody* are of type <et,t>. For two place predicates, *everybody* and *somebody* are of type <eet,et>. Thus for a simple sentence such as *somebody offended everybody*, this approach would dictate that we use the two place predicate denotation for *everybody*, which immediately combines with *offended*(type <e,et>) by functional application, and then combine the resulting constituent with the one place denotation of *somebody*. Using this approach, we can adequately derive the truth conditions under which the sentence *somebody offended everybody* is true. These denotations are inadequate in capturing the truth conditions of a sentence like *somebody offended every linguist*. The moment that we tack on an NP to the determiner, we get a construction that results in a type mismatch. The solution proposed must then additionally modify the denotations for everybody and somebody. At this juncture, instead of simply proposing another denotation, Heim and Kratzer decided to propose a general rule by which denotations that result in type mismatches can be transformed into the contextually adequate type. Part of the theoretical justification for finding general principles for type shifting is because linguists don't want multiple denotations to be stored in the lexicon for each quantifier. A principle merit used across the board in judging the merits of a linguistic theory rely heavily on the concept of parsimony. We want to minimize a theory's set of axioms while maximizing its explanatory breadth. Under a type shifting account then, the theory only needs one piece of information to be stored in the lexicon of each individual quantifier: the bedrock denotation <et,<et,t>>. From this denotation, the other piece of information that needs to be stored is the proposed type shifting rule. From these two pieces of information, it is possible to derive all further denotations dependent on what different linguistic contexts require. The type shifting rule proposed in the book takes the original dentition for every and some of type <et,<et,t>> and transforms it into a meaning of type <et,eet,et>. It is important to note that the effects of type shifting rules do not percolate to syntactic or phonological domains. It prefaces the type shifting rule with the understanding that both denotations are syntactically identical and homophonous(meaning same sound different meaning). This theory is therefore only operational when restricted to the purview of semantics. In constructing the type shifting rule, the value description for [[$\boldsymbol{\delta}_2$]] takes [[$\boldsymbol{\delta}_1$]] and inputs the set of functions and individuals that returns [[$\boldsymbol{\delta}_1$]]’s value description. The modification that [[$\boldsymbol{\delta}_2$]] makes is inserting a lambda term of type <e,t> in the set of functions it inputs into [[$\boldsymbol{\delta}_1$]]. On a surface level, type shifting rules of this sort seem to be nothing more than an ad hoc theoretical pivot.While type shifting rules may seem overly complex at times, it's important to note that the concept of type shifting isn't without solid theoretical justification. Taking the semantics of [[and]] and [[or]], it is easy to intuitively understand the basis for type shifting rules. The simplest denotation for [[and]] and [[or]] is <t,<t,t>>. These conjunction denotations fail, however, when they connect phrases that by themselves don't return truth values. In instances where coordination is involved such as “Sam will be at school or at home" neither of the inputs into [[or]] are of type t. Additionally, given the sheer number of different items and lexical categories that could be used alongside coordination, it is clear that these denotations could not be listed separately in the lexicon. This combinatorial explosion is worsened when we notice that we can recursively embed "and" and "or" phrases and use coordination for certain phrasal constituents and not for others. It would simply be computationally intractable from a point of view of the constraints of working memory on biological computation to try to come up with all the denotations for "and" and "or". This is a case, therefore, where it is intuitively obvious why type shifting rules are of theoretical necessity: whether this property holds for determiners is questionable.

Heim and Kratzer's other proposal to quantifier ambiguity minimizes the rule set on the semantic side, limiting the compositional rules to only functional application, predicate modification, and predicate abstraction. Crucially, there are no type shifting rules. Type shifting rules are semantic operators that allow the quantifier to remain in the in situ position in some cases by type shifting to the appropriate type. This appropriate type is one that permits the quantifier to combine through functional application with the appropriate neighboring words. Type shifting rules will not concern us in this paper, however. An initial question that arose in assessing how quantifiers undergo movement is whether the entire DP housing the quantifier undergoes movement. If we assume that the movement operations constraining quantifiers parallel movement operations constraining wh-movement, for example, we would expect locality constraints to be obeyed. We will look at cases where quantifiers take narrow scope and whether the possibility of quantifier adjunction in the VP is plausible. Many examples of this form involve auxiliary negation. Take the sentence "Al didn't attend more than two meetings". There are two interpretations of this sentence where negation takes wide scope over [more than two] and where [more than two] takes wide scope over negation. Illustrated by a syntactic tree, the scopal reading with negation taking wide scope places [not] at the functional projection site of I, with [more than two meetings] located lower in the tree at the DP site at the specifier of VP. Since this tree places the DP at the specifier of VP, this reading commits us to the idea of VP-internal subjects. Reading off this tree we obtain the interpretation that Al attended no more than two meetings, attending either two, one or zero meetings. The syntactic tree with [more than two meetings] taking wide scope places this phrase at the DP node in the specifier of IP and [Not] is located below in the tree at the specifier of the I-node directly above the VP. Reading off this tree we get the interpretation where Al did not attend more than two meetings, possibly missing three, four, or fifty meetings. While VP adjunction may not have been required in the above example, Heim and Kratzer came up with a sentence whose interpretation requires it: "Al didn't return every clock to its owner". One interpretation of this sentence is where he did return some clocks to their owners, just not every clock. Another interpretation of this sentence is where he still has every clock, he hasn't returned a single clock to its owner. In the tree where [did not] takes wide scope, arriving at the specifier of I', the DP every clock is left to either stay in its in-situ position or move to the specifier of VP position. Since every clock is bound by "its", the constituent must move to a position where it can dominate its own trace and the pronoun its. For [every clock] to be able to dominate its own trace, however, it must move out of its in-situ position. Thus the only slot left unoccupied for movement is the site of VP adjunction. This example brings credibility to the idea of VP adjunction, as the only way to derive one of the two interpretations of this sentence requires it. Now that we have seen examples where QP's quantify with VP's, we can look at other examples with quantifying in PP's. Consider the sentence "No student from a foreign country was admitted". The desired interpretation of this sentence is one where of all the students admitted, none of them was from a foreign country. If we try to move up the embedded PP [a foreign] to the specifier of IP position, we have to leave a trace inside the constituent DP [no student from t] which is located at the specifier of the lower IP node. The interpretation of this tree, however, does not match with the desired meaning as this tree can evaluate to true if there is just a single foreign country that was represented in admission: students from other foreign countries could be admitted. If we were to try an alternative route, say place the DP [no student from t] at the specificer of the first IP and [a foreign country] at the specifier of the second IP we run into additional problems. The trace in the first constituent [no student from t] is unbounded by the anaphor it represents. This is a direct violation of the criterion that states there can be no unbounded traces. The most obvious direct solution would be to carry the rest of the PP constituent [a foreign country] up to the specifier of the first IP position. In this tree, the DP located at the specifier of IP consists of [no [NP student [PP PRO [PP [DP a foreign country] [PP t1 from t2]]]]. This tree resolves the problem with traces above as both traces are c-commanded by their respective antecedents. While this tree does present the desired meaning, it commits us to the fact that "PP modifiers are sometimes allowed to rewrite as clausal structures"(Heim & Kratzer). We have now seen quantifier adjunction to VP and PP. Next I will present an example that illustrates the need for adjunction to DP. The following sentences make the case nicely, "John met neither a student from every class nor a professor". The meaning we want our tree to represent is the state of affairs in which John did not meet any students from any classes and that he didn't meet a professor. The constituent [neither-nor] functions to connect to different quantificational DP's in this sentence such as [a student from every class] and [a professor]. The tree that yields the desired meaning requires the DP node at the specifier of IP to do a lot of heaving lifting. This DP node contains both quantificational DP's, with a trace embedded in the DP [every class a student from t]. The constituent [neither-nor] is located higher up in this DP relative to the other two quantificational DPs. This way, both DP's are conjoined with neither nor ranking highest in the c-command hierarchy of the DP node. The meaning of this DP then gets transferred to the IP [John met t1] after intersecting with the intervening 1 in the tree. This way the negation property of neither-nor can apply to both DP's through its c-command relation in much the same way as [it-is-not-the-case -that] would have functioned had it appeared in the sentence. Take the sentence "two politicians spy on someone from every city". This sentence contains three quantifiers and would therefore be expected to have more readings than usual. Two scopal readings that yield the correct interpretation are [every > someone > two] and [two > every > someone]. The scopal reading where every is the only quantifier to undergo a movement operation, to the highest node in the read, is [every > two > someone]. This scopal reading, however, yields the wrong interpretation. Under this reading, since every takes two and someone as the distributed shares, there could potentially be two different politicians per city. Additionally, politicians from the same city could be spying on different people. The fact that this last scopal reading leads to the wrong semantic interpretation tells us that quantifiers are constrained by the DP's they find themselves in. By this, I mean that a quantifier can at most move to the edge of its DP. While this constraint is satisfied in the first two readings, and thus the meanings are conserved, this fact is not upheld in the last reading. Sentences of this kind present some evidence for the existence of adjunction to

DPs. If, however, we try to apply DP adjunction to a sentence like [someone from every city despises it], we run into a problem with unbounded pronouns. If the DP [every city [someone from t1]] were to undergo adjunction to DP, this would leave the VP as [despises it1]. The problem here is that the city would only c-command the trace located inside of the embedded DP and not the pronoun "it". More formally, this fact is termed as the island constraint and applies to constructions of the form [NP PP]. This is the same as the island constraint applying to wh-movement, which is theoretically desirable given that we want all movement operations to generally follow the same set of principles. If we are to maintain our proposition of DP adjunction for inverse readings, this would require us to make the seemingly unwarranted move of radically altering existing theories of pronominal anaphora. Next I will examine properties of quantifiers and their interactions with intensional predicates such as want or need. Taking the sentence "Sam believes someone is out to get him" there are two distinct interpretations. On the de dicto reading, there is not any specific individual that Sam knows is out to get him: it just reflects his general paranoia. On the *de rea* reading, the quantifier someone refers to someone specific in the real world. Take the following two sentences:

(a) Max needs [a lock of mane from every unicorn in an enchanted forest]

(b) Felix wants [a story about every dogcatcher in a large California city]

On the de dicto reading of (a), there could be multiple locks of mane and enchanted forests as long as hair is taken from each unicorn in the particular chosen forest. Similarly, the de dicto reading for (b) where multiple values could fill the criterion of a large city in California and story about the dogcatchers(i.e. multiple cities and multiple stories). What must be upheld is that from whichever city is chosen, every dogcatcher from that city gets a story. These sentences both require that...

Total locks of mane = |{ u | u is a unicorn from enchanted forest f }|

Total stories = |{ d | d is a dog catcher from big California city c}|. Note that the left side set depends entirely on the length of sets on the right, where the variables depend on the particular forest and the particular city c chosen. The interesting fact about these sentences is that although locks of mane, stories, forest and city are all read de dicto, only the latter two nouns receive broad interpretation: as referring to the set of all enchanted forests and big California cities respectively. The fact that both NP's are read de dicto, and interact in the fashion described above, means that quantification over them must take place before they combine with the intensional verb. To do so requires the inverse reading of the quantifiers embedded in the NP such that the reading of the NP portion of the tree would be [an enchanted forest [every unicorn in him1[a lock of mane from him0]]]. This tree formation gives enchanted forest scope over every and a lock of mane, reflecting its status as the broadest de dicto interpreted noun in the NP. Moreover, this inverse ordering reflects the correct ordering in predicate logic: $\lambda R \exists z$[en-forest'(z) & $\forall y$[unicorn(y) & in(y, z) -> $\exists x$[l-of-m(x) & from(x, y) & R{x}]]. Put in words, this equation corresponds to there being a forest, such that for every unicorn in it, at least one lock of mane is needed from each.

Next I will be delving into the Cooper Rider Storage account of quantifiers. Doing so will let me present an alternative solution and syntactic tree for the sentence "someone from every city despises it". While the tree presented by Heim and Kratzer, as discussed above, allows for the composition of the correct semantic interpretation and allows for "it" to be bound by "every city”, it fails to respect the existence of island constraints on DP movement as gleaned from the case with three quantifiers.

The storage account is particularly unique in that it doesn't create multiple syntactic trees for different interpretations. Instead, the trees are winnowed down into a single one, and different scopal readings are handled by two operations called quantifier storage and quantifier retrieval. Take the sentence “every man loves some woman” for example. The single tree structure representing this tree parallels the surface form ordering. The two interpretations of this sentence are one where there exists a woman such that every man loves her or where each man loves some woman separately. More formally, this is captured by these two equations: $\exists y \forall x Love(x, y)$ narrow scope

(1) : $\exists y \forall x Love(x, y)$ narrow scope

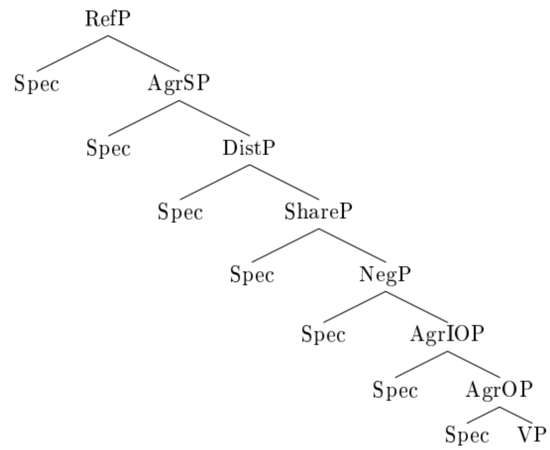
(2) : $\forall x \exists y Love(x, y)$ wide scope

In the Storage account, the tree is interpreted from the bottom up. NP’s lower on in the tree can either be directly combined with elements farther up or can be stored. If we choose not to store “some women” when we reach the VP love in the tree we receive the narrow interpretation of the sentence. If, on the other hand, we choose to store the NP it stays stored until it gets “quantified in” at the S node. Another term used here is “returned”. The resulting scope when the NP get quantified in at the S node is wide scope. Island constraints are upheld by the fact that stored quantifiers are limited from moving upwards in the tree past their islands. The operator which limits such movement is known as the INT, which identifies islands in terms of a set of syntactic nodes I. I in this case stands for the set {NP,Q,R}. The function INT forces all quantifiers to be stored out before the interpretation of the island node can be combined with other constituents.

Now back to the sentence: “someone in every city despises it”. The VP [despises it] is lowest in the tree and therefore gets interpreted first. The next constituent NP to get interpreted is [every city]. Now at this juncture we can choose to either store or not store [every city]. If we choose to not store it, [every city] would stand alone not linked to someone. This would yield the interpretation where there is someone who is simultaneously in every city at once, and despises all of them. This sentence is doubly strange when we realize that the singular marker “it” refers to every city at once. Instead, if we store [every city] we then have to combine it with the proposition “in” to complete the PP node. Then we combine it with the interpretation of the second NP node [someone]. At this point we run into some problems. Due to island constraints we cannot combine [someone is {{every city}}] with despise (from now one {{}} refers to a stored item). If we were to quantify out, every city would only have scope over the NP constituent [someone in every city]. This is problematic, however, as doing so would prevent binding with the pronoun “it” in the lower VP. Assuming Larson’s proposal of embedded storages, where stores of quantifiers can be stored within themselves, we would store the entire NP [someone in every city] at the juncture where it combines with the VP. Since the broader NP is not in violation of island constraints, it is able to freely combine with the VP. Taking this route, the NP [every city] remains stored, and yet when embedded inside of another stored NP does not violate island constraints enforced by INT. At the point where the retrieval process takes place, both the broader stored NP is retrieved in addition to the other NP stored within the store. The Storage approach in general solves the problem of binding pronouns in the inverse readings of quantificational phrases with “subjacent character of quantifier scope”(Larson, 1985)

The problem with quantifiers taking different scopes is not merely that there are multiple, different scopal readings for a given sentence. Some sentences only permit a subset of all possible scopal readings if we were merely taking a quantifier raising approach into account. Thus for a sentence with n quantifiers, it is not necessarily the case that there are n! possible scopal readings of the sentence. Certain quantifier interactions “exhibit a smaller set of inverse scopal options than would be predicted if QR applied to them”(Ben-Shalom, 1993). To determine this smaller set of inverse scopal options, we must appeal to the type of the quantifier which is dependent on how a quantifier behaves in a broad class of different linguistic environments. A solution proposed by Szabolcsi (1997) states that quantifiers perform feature checking operations on functional projections located higher up in the tree. Importantly, these functional projections are “ quantifier-specialized”. I will be examining this proposal first, moving to explain some of its shortcomings and looking at alternatives.

The most rudimentary treatment of quantifiers proposes that quantifiers are semantically blind. By this, we mean that irrespective of what the quantifier actually means, determinations of scope are dependent solely on c-command relations in a tree. Thus if a quantifier c-commands a phrase, then its scopes over that phrase. In this conception, quantifiers of certain types do not have specific landing sites. In Beghelli and Stowell’s (1994) paper, they argue strictly against this approach by proposing a set of functional projections that do discriminate on the basis of different quantifier types. The order of the different functional projections is based on the hierarchy of quantifiers which determines which types of quantifiers can take scope over other quantifiers. This proposal rests at the syntax-semantics interface. While the movement operations of quantifier raising are syntactic in nature, the determination of the hierarchy of functional projections is based on semantic considerations.



The order of functional projections is as follows: (1)RefP: Referential Phrase
(2)AgrSP: Subject Agreement Phrase (3)DistP: Distributive Phrase (4)ShareP: Distributed Share Phrase (5)NegP: Negative Phrase (6)AgrIOP: Indirect Object Agreement Phrase (7)AgrOP: Direct Object Agreement Phrase (8)VP: Verb Phrase

For a quantifier to acquire its scope, it must move to the head of the functional projection that it agrees with. Definite phrases, such as “the 3 dogs” move to the highest functional projection in the hierarchy. This makes intuitive sense, as the phrase cannot act as the distributed share for distributive phrases because the phrase isn't a variable that can be satisfied by taking multiple, different arguments. On the other hand the indefinite phrase “3 dogs” can act as the distributed share for the distributive operator, as the phrase [3 dogs] (can also be written as 3 of the dogs) is a variable phrase that encompasses a set of many three-dog triples. Adding [the] to [3 dogs] trims that vast set to size of one, namely the single group of three dogs the salient discourse context selects. Operators such as every cannot take scope over unitary phrases such as [the 3 dogs] as there aren't multiple elements in the set that every can internally target. While definites are barred from moving to the ShareP projection, indefinites can move to both ShareP and RefP. Although definites can never move to ShareP, they can nevertheless sometimes be interpreted as a distributed share phrase. These instances are called pseudo-distributivity and are only possible by a covert movement of an invisible operator, each. If we take the sentence “two of the men read three books”, [three books] can move to RefP and become referentially independent of “two of the men”. (*)Two of the men, however, is not referentially dependent on [three books] because there is no distributive operator that exists between them to license such an interpretation. Referentially dependent constituents have meanings that must be calculated in regards to the additional constituents that they depend on. If we were to slightly modify this sentence such that it read “two men read three of the books”, [three of the books] could be distributed over two men because each would appear covertly before the phrase [three of the books]. Thus the sentence would read [RefP two men [AgrSP t1 EACH [ShareP three of the books [...]]]]. Each serving as the distributive operator, to distribute the meaning of [three of the books] to the [two men], now licenses the interpretation where six books are read in total as each man reads three books separately. This case also illustrates the idea of referential dependency, as the correct number of books can only be calculated with regards to the number of men there are. Had there been three men, there would need to be nine books total. (*)The silent operator can occur below all functional projections except RefP where its functional projection is filled. This is the reason why [two of the men] could not be distributed back over when the direct object [three books] appeared in RefP. In sum, definites and bare definites display distributivity properties but only when each appears, not phonologically realized.

This account of quantifiers over and under generates in certain instances. Take the sentence “four students read three books”. Both scope readings of this sentence exist. This account states that “object bare numeral indefinite” can never undergo movement over the subject position and that they do not reconstruct to their base position. Thus this account only generates the direct scope reading and therefore undergenerates for this example. An example sentence where their account overgenerates is the following “less then four students read exactly three books”. Since they say that modified numeral indefinites are able to reconstruct back to VP, we would expect the inverse reading to be available. This is not the case, however, as [less that four students] cannot work as the distributed share back to [exactly three books] when it moves to the subject position.

Compared to other proposals, methods that create different functional projections for quantifiers don't run into pronoun binding problems as other accounts do: such as quantification into S, VP, DP or PP. This added theoretical surplus, however, is offset by the fact that this system largely breaks down when a single sentence has multiple instances of the same quantifier. Since functional projections do not disambiguate between multiple instances of the same quantifier in different positions, both quantifiers are said to move to the same position. This would mean, however, that no quantifical hierarchy exists between those same two determiners. The only source of retrieving the hierarchy would be to look lower down in the tree at the constituents that contain these quantifiers as traces. Whichever trace appears before the other trace corresponds to the anaphorically linked outscoping quantifier. However, given that our formal system operates like a machine, moving from the topmost node to the bottomost, it is blind to the disambiguating trace ordering until it reaches that point in the tree. This leads us to the strange conclusion that scoping relations between quantifiers is unknowable until binding operations are finalized. This would be particularly strange for a method that seeks to put quantifier scope hierarchies at the theoretical, and literal in the tree, forefront.

Of all of the models I have laid out for dealing with quantifier scope ambiguity, I find the Storage approach to be most compelling. Specifically, the sum theory of Storage after Larson tacked on an additional few amendments. Not only is this in keeping with the general approach of generative grammar, that relies on movement operations over syntactic structures as opposed to type-shifting rules, but this approach solved the anaphor binding of pronoun problems present in mostly all other proposals involving movement.

Sources:

[1]https://s3.amazonaws.com/academia.edu.documents/46690795/Szabolcsi_Strategies_for_scope_taking.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1556474434&Signature=KzHVcuRrmYUGkbnaXVevUMtFu6c%3D&response-content-disposition=inline%3B%20filename%3DStrategies_for_scope_taking_1997.pdf
[2]http://www.cssp.cnrs.fr/eiss5/eiss5.pdf#page=215
[3]http://scholar.harvard.edu/files/yxiang/files/ho14-quantifier_raising_and_scope_ambiguity.pdf
[4]https://www.cs.brandeis.edu/~ling130/s14-QR.pdf
[5]http://linguistics.berkeley.edu/~jenks/Research_files/QuantifierRaising_Feb2013.pdf
[6]http://www.gist.ugent.be/file/218
[7]https://www.ling.uni-potsdam.de/~zimmermann/teaching/Syntax-Semantic_Interface/Presentations/May-Presentation.pdf
[8]http://www.bu.edu/linguistics/UG/course/lx502/_docs/lx502-quantifier%20raising.pdf

Figure 1