Modelling Cross-linguistic Variation in Binding Using Synchronous Tree Adjoining Grammar

Dennis Ryan Storoshenko and Chung-hye Han
Yale University and Simon Fraser University

1. Introduction

When used as bound variables, English pronouns such as *her* must have a certain minimum syntactic distance from their c-commanding antecedents:

(1) a. * Every girl$_i$ loves her$_i$.
b. Every girl$_i$ loves her$_i$ father.

As shown in (1), an English bound variable pronoun can appear within the same clause as its antecedent, but they cannot be co-arguments. Conversely, Korean *caki*, which can be analyzed as a bound variable (Han et al., in press), does not have such a restriction:

(2) a. Motwu$_i$-ka caki$_i$-lul salang-ha-n-ta.
everyone-NOM self-ACC love-do-PRES-DECL
'Everyone loves himself.'
b. Motwu$_i$-ka caki$_i$-uy appa-lul salang-ha-n-ta.
everyone-NOM self-GEN father-ACC love-do-PRES-DECL
'Everyone loves his father.'

Further, for both *caki* and the English pronouns, binding across clauses is unbounded:

(3) a. Every girl$_i$ knows [that someone believes [she$_i$ is intelligent]]
b. Motun sonye-nun [salamtul-i [caki-ka ttokttokha-tako] sayngkakha-n-tako] mit-nun-ta.
every girl-TOP people-NOM self-NOM intelligent-COMP think-PRES-COMP believe-PRES-DECL
'Every girl believes that people think that she is intelligent.'

These examples show that the variable binding may obtain across two clause boundaries, though there is no limit as to how far this dependency may be stretched. Thus, the contrast which needs to be defined here is a fine gradation of locality. In this paper, we show that by modelling the binding relation within the Synchronous Tree Adjoining Grammar (STAG) framework, we can express this contrast as a constraint on derivations making use of bound variables. Furthermore, the contrast is formulated in such a way as to predict a limited set of possible locality binding constraints, and thus a limited typology of bound variables.

In Section 2, we introduce the key concepts of Synchronous Tree Adjoining Grammar upon which our analysis is based. Then, in Section 3, we illustrate the crucial difference between the English and Korean derivations. Section 4 abstracts from the observations in the derivations, and argues for the definition of binding locality constraints as a function of derivational properties, rather than using the derived syntactic (or semantic) structure. This will ultimately result in a system where constraints on all three of the derived syntax, the derived semantics, and the derivation itself will work in concert to capture the observed facts. Finally, Section 5 summarizes our analysis and forecasts future extensions.

2. STAG Basics

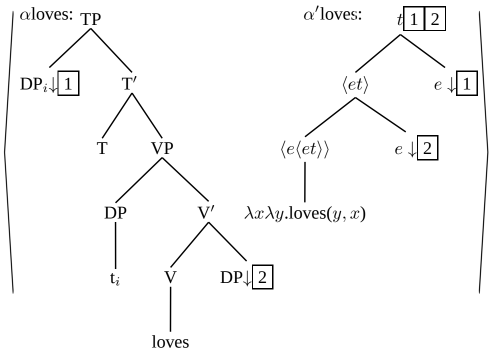
At the core of this formalism is Tree Adjoining Grammar, first formalised in Joshi et al. (1975), with the first major linguistics application coming in Kroch & Joshi (1985). In its barest foundation, TAG is a tree-rewriting system which has as its units elementary trees, and two operations, Substitution and Adjoining (to be defined below), through which elementary trees are combined. As applied to linguistics, a lexicalised TAG is a TAG in which each elementary tree has one and only one lexical item. In Frank (2002), a TAG-based system of syntactic analysis is presented, showing that elementary trees built upon GB/Minimalist principles can be combined using the TAG operations to formulate a workable model of syntax with equal or greater explanatory power than Minimalism alone. A key component of this system is the Condition on Elementary Tree Minimality, which states that for a given lexical item, its elementary tree will contain all the necessary functional projections to support the arguments and other local dependencies of that lexical anchor.

Synchronous Tree Adjoining Grammar, or STAG, (Shieber & Schabes, 1990; Shieber, 1994; Nesson & Shieber, 2006) builds upon TAG syntax by deriving both syntactic and semantic trees in parallel. In STAG, each lexical item will in fact have two trees, one for syntax and one for semantics. The syntax tree is again built upon GB/Minimalist principles, while the paired semantics tree expresses a typed lambda calculus representation of the lexical item and its argument structure. To illustrate the system, we will derive a very simple sentence:

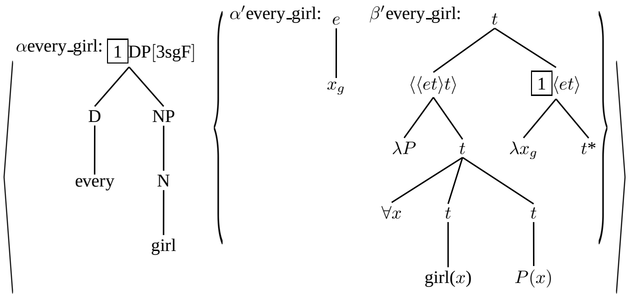
(4) Every girl loves Bill.

Sample elementary tree pairs for the predicate *loves*, the quantifier *every girl*, and *Bill* are given in (5), (6), and (7) respectively.

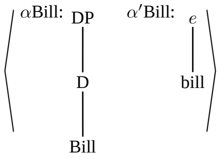
(5)



(6)



(7)



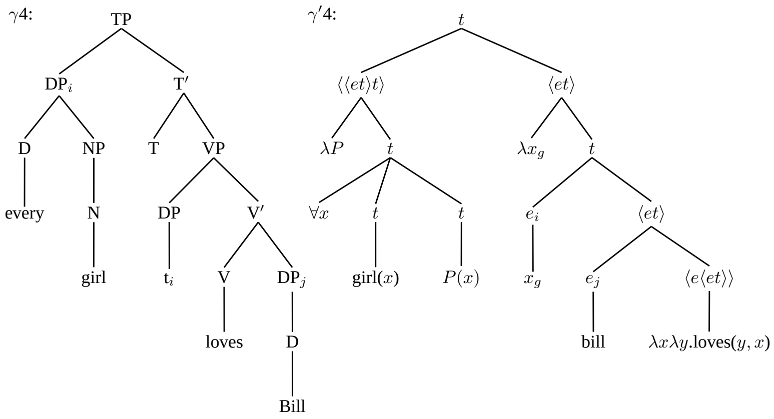
Looking first at the *loves* trees, we find a very simple syntax, consisting minimally of VP and TP projections. This tree could be updated with a more contemporary vP structure, but such details are irrelevant to the issue at hand, and are eliminated in the interests of space and clarity. We do illustrate the VP-internal subject hypothesis, and subsequent movement of the subject from [Spec, VP] to [Spec, TP], as an example of the kind of overt movement permitted within TAG syntax: movement is possible, but only within a single elementary tree. Most important to note are the argument positions in this tree. The two DP nodes are marked with ↓, indicating their status as substitution sites. Substitution is one of the two TAG combinatory operations available for elementary trees; nodes marked as substitution sites can be seen as open ‘slots’ where an elementary tree whose root node matches the node label of the substitution site may be inserted. In this case, there are slots for two DPs, representing the arguments of *loves*. These nodes are also marked with boxed numerals, indicating links between the syntax and semantics trees. Whenever a TAG combinatory operation takes place at a linked node, members of the same elementary tree pairs must be combined at those linked nodes in the syntax and semantics. In this case, this ensures that the subject and object will be converted by the correct lambda expressions.

Turning to the tree pair for *every girl*, the syntactic representation is a simple DP. On the semantics side, the generalized quantifier structure is represented as a multi-component set (MCS). This treats the quantifier as having two distinct parts: a variable part (α'every_girl) which will substitute at the linked argument position of the syntactic DP, and a scope part (β'every_girl) which combines with the predicate by way of the second TAG combinatory operation, adjoining. Looking at (β'every_girl), note that it is a recursive structure: its root node is of the same type as one of its leaf nodes, t. This recursive structure, or auxiliary tree, can effectively be spliced into another elementary tree at a matching node. More formally, an adjoining site is defined within the target tree, and the target tree is split into two sub-trees (top and bottom) at that site. The auxiliary tree is inserted at the adjoining site along the frontier of the top sub-tree, and the bottom sub-tree then attaches to the recursive leaf node on the auxiliary tree. In this case, the scope part of *every girl* targets the root t node of (α'loves), adjoining into that node, and defining the scope of the quantifier.

Lastly, *Bill* is represented as a DP in the syntax, and a simple entity of type e in the semantics. These two trees will also substitute at the linked object positions in the *loves* trees. Han et al. (2008) provides an alternative account of proper names, also treating them as generalized quantifiers, but that level of detail is not needed for this simple example derivation.

Once all combinatory operations are complete, the end result is a pair of derived trees:

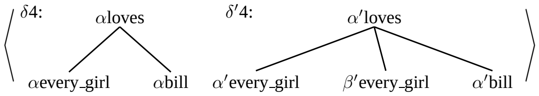
(8)



The tree on the syntax side gives the syntactic constituent structure and linear order of elements, while the semantics tree can be used to calculate the meaning of the sentence.

TAG combinatory operations are recorded in separate derivation trees:

(9)



In these derivation trees, each node represents an elementary tree. Dominance relations in the derivation trees encode the directionality of the combinatory operations: elementary trees are dominated by the trees into which they adjoin or substitute. Crucial to the STAG formalism is the restriction that the derivation trees on both the syntax and semantics sides be isomorphic. That is, the same derivational steps must be used in both parallel derivations. In cases where there is a mismatch between a syntactic and a semantic MCS, such as *every girl* having only one syntactic tree, but a two-component semantic MCS, exact isomorphism is not possible. In these cases, derivation trees must be maximally isomorphic, meaning that wherever possible, isomorphism is respected.

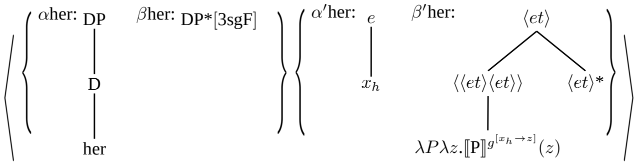
Also note that each member of the MCS is represented as a distinct node in the derivation tree. When an MCS is constrained to combine locally, all components must combine with the same elementary tree. In terms of the derivation tree, this means the MCS nodes are always sisters, which we see for (α'every_girl) and (β'every_girl) in (9). However, Chiang & Scheffler (2008) introduces the notion of delayed locality for MCS combination. Under delayed locality, a derivation is licit so long as there is at least one derivation tree node which dominates all members of a MCS. The delay of a derivation for a particular MCS is defined as the set of derivation tree nodes along a path from one member of the MCS to the other, including the MCS members, but excluding the common dominating node. Thus, for any given MCS, where n is the cardinality of the MCS, d, the cardinality of the delay, will always be at least n, as is the case for the quantifier on the semantic side of (9). Once an MCS combines via delayed locality, the d value for that MCS's delay will be at least n+1, which we define as the threshold value for delayed locality. Following Nesson & Shieber (2009), the value of d can then be employed in defining constraints on a derivation; in the next section we present our STAG analysis for bound variables, re-interpreting the observed locality facts in terms of derivational delays rather than in terms of the derived syntax.

One of the virtues of TAG-based syntactic analysis is that the elementary trees provide a natural definition of a domain of locality. Indeed, much work of the past 20 years has shown that apparently long-distance dependencies such as raising and *wh*-movement, among others, can be reduced to local dependencies interacting with recursive structures introduced by way of adjoining. It is for this reason that the analysis of bound variables has remained a challenge within the TAG literature, as variable binding represents a long-distance dependency which cannot be so simply reduced. Couched in the "traditional" definition of local as referring to a clause-bounded domain, the grammatical (1b) and its Korean counterpart (2b) are local in that they represent a relationship within a single clause. However, the variable and its antecedent are arguments of distinct predicates, as *every girl* is the subject of *loves*, while *her* is in fact an argument of the possession relation contained within the DP *her father*. Recalling that elementary trees are defined based on argument structure, and that locality is defined based on elementary trees, it then falls out that in TAG-locality terms, these examples are non-local. In TAG terms, there is no easy way to reduce the observed antecedence relation to a local dependency, and it is for this reason that we need to move into the realm of delayed locality as we consider bound variables. Ultimately, we will exploit this difference in TAG locality as the key factor in distinguishing the English and Korean facts which show contrasting behaviour within what are generally considered to be equivalent local domains.

3. Modelling English and Korean Binding Facts

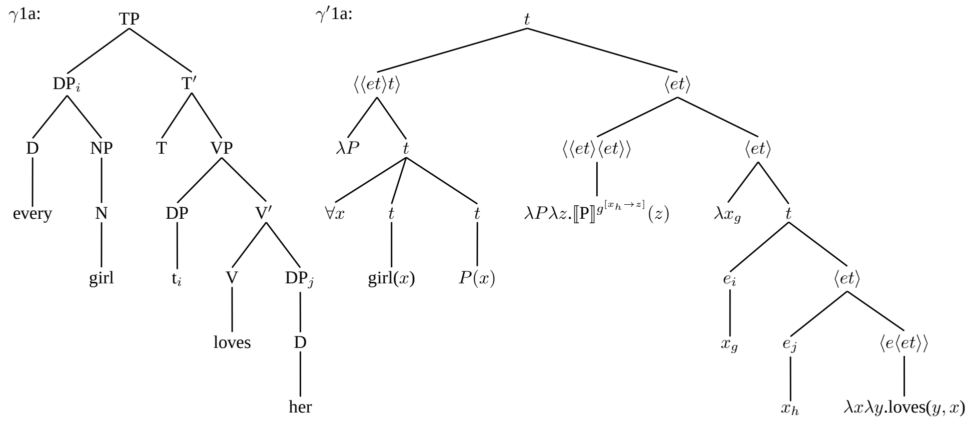
In the previous section, we introduced STAG elementary tree pairs for both *loves* and *every girl*. In order to attempt to derive the ungrammatical (1a), we need only introduce the elementary trees for the bound variable *her*:

(10)

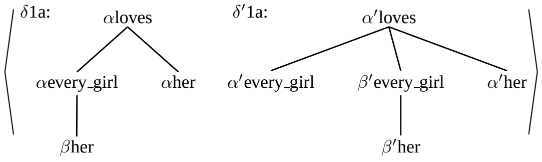

αher: DP
D
her
βher: DP*[3sgF]
α'her: e
x_h
β'her: $\langle et\rangle$
$\langle\langle et\rangle\langle et\rangle\rangle$
$\langle et\rangle$*
$\lambda P\lambda z.[\![\mathrm{P}]\!]^{g^{[x_h\rightarrow z]}}(z)$

The bound variable is represented as a MCS on both the syntax and semantics sides. For both, there is a component which will substitute at the relevant argument position, and an auxiliary component which will adjoin into the antecedent. In the syntax, the auxiliary component, a "defective" auxiliary tree consisting of a single node, carries the *phi*-feature specification of the bound variable, ensuring agreement between the variable and its antecedent. In the semantics, the auxiliary component will adjoin into the scope part of a generalised quantifier at a linked node of type ⟨et⟩, carrying an instantiation of the Binder Index Evaluation Rule from Büring (2005). This mechanism is what allows a quantifier to bind the new variable. Because the two MCS components necessarily combine with different elementary trees, the derivation must make use of delayed locality. Derived and derivation trees for (1a) are given in (11) and (12), respectively:

(11)


γ1a:
TP
DP$_i$
T$'$
D
NP
T
VP
every
N
DP
V$'$
girl
t$_i$
V
DP$_j$
loves
D
her
γ'1a:
t
$\langle\langle et\rangle t\rangle$
$\langle et\rangle$
λP
t
$\forall x$
t
t
girl(x)
$P(x)$
$\langle\langle et\rangle\langle et\rangle\rangle$
$\lambda P\lambda z.[\![\mathrm{P}]\!]^{g^{[x_h\rightarrow z]}}(z)$
$\langle et\rangle$
λx_g
t
e_i
x_g
$\langle et\rangle$
e_j
x_h
$\langle e\langle et\rangle\rangle$
$\lambda x\lambda y$.loves(y,x)

(12)



Using the elementary trees as formulated, there is nothing in the derived trees to indicate why the sentence is ungrammatical. The syntactic form is easily derivable, and carrying through the calculations on the derived semantic tree yields the expression in (13):

(13) $\forall x$[girl(x)][loves(x, x)]

This appears to provide the intended bound variable semantics. With no indication of any problems in the derived trees, we turn next to the derivation trees.

As before, the derivation trees are maximally isomorphic. In considering the delays, we will restrict our attention to the semantics side of the derivation where all the delays inherent in the derivation may be observed. Here, we find that there are two delays, one for the quantifier *every girl* and one for the bound variable *her*:

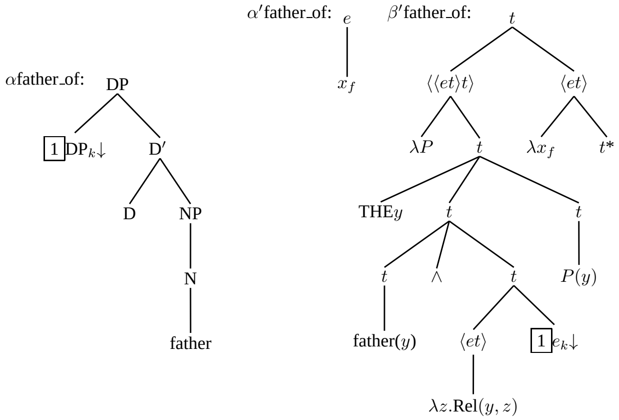
(14) a. Delay for *every girl* (Semantics Side)
$\{\alpha'$every_girl, β'every_girl$\}$: d=2

b. Delay for *her* (Semantics Side)
$\{\alpha'$her, β'every_girl, β'her$\}$: d=3

Here, we see that while the bound variable has indeed combined by way of delayed locality, it has done so using the most minimal delay possible. Given that this is a two-member MCS, d=3 represents the threshold for delayed locality.

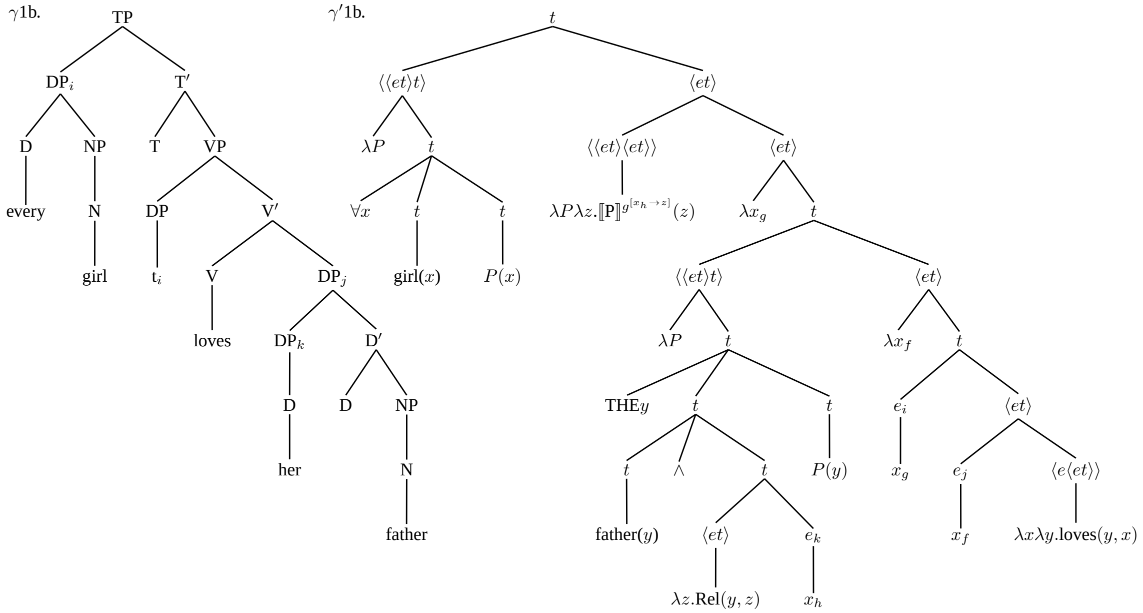
To compare this with the derivation of the grammatical (1b), we first need to introduce the elementary trees for *father_of*, which we have already hinted will contain its own argument structure, representing the possessive as a two place Rel(ation):

(15)

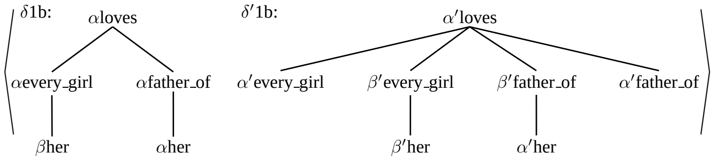


Again, this is defined as a generalized quantifier structure, and as with the scope part of *every_girl*, the auxiliary tree on the semantic side here is recursive on type t. As such, both quantifiers will essentially be "competing" for the same adjoining site. This is a desirable situation, as it more generally allows for quantifier scope ambiguities to be simply derived in STAG. Where two scope-taking elementary trees are competing for the same root node, either ordering is possible, meaning that either quantifier can outscope the other within a single clause, regardless of their syntactic positions. In this way, ambiguities normally attributed to unordered quantifier raising are mirrored in STAG. However, only one of the two potential orderings is viable when variable binding is added to the picture:

(16)



(17)



Completing the semantic calculation on the derived tree in (16) yields the formula in (18):

(18) $\forall x$[girl(x)][THEy[father(y) $\wedge$ Rel(y,x)][loves(x,y)]]

Here, the variable binding has yielded the correct formula. Had the quantifiers adjoined in the opposite order, the end result would still converge as a calculable derivation, but it would leave the x_h variable unbound, as (β'her) would be lower down in the derived semantic tree than the (α'her) component substituted into (β'father_of). We assume this derivation to be blocked by a more general constraint against unbound variables at the end of semantic composition.

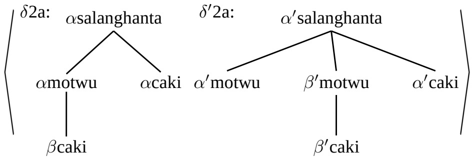
Once again limiting our attention to the semantic derivation tree in (17), we now find three delays:

(19) Delay for *every_girl* (Semantics Side)
$\{\alpha'$every_girl, β'every_girl$\}$: d = 2
Delay for *father_of* (Semantics Side)
$\{\alpha'$father_of, β'father_of$\}$: d = 2
Delay for *her* (Semantics Side)
$\{\alpha'$her, β'her, β'father_of, β'every_girl $\}$: d = 4

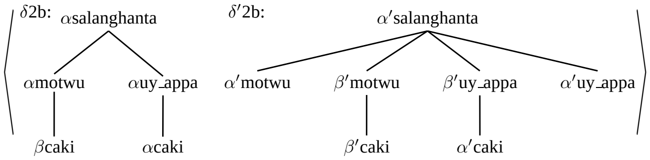
The quantifiers behave exactly as in the earlier examples, but the d value for the bound variable MCS is now 4.

For the Korean examples, while the internal details of the elementary trees are different, the derivations exactly match what we see for English:

(20)



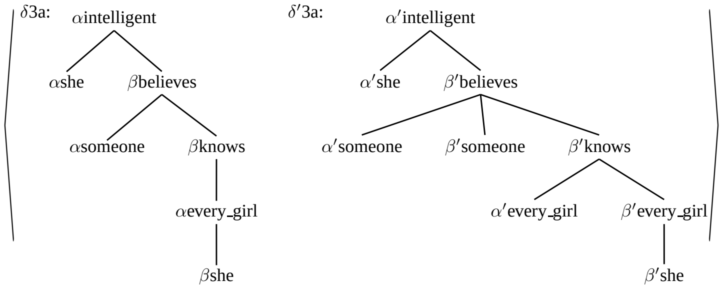
(21)



As with English, the difference here is in the d value for *caki*, 3 in the case of (20), but 4 in the case of (21). However, the contrast is that both are grammatical for Korean.

In closing this section, we briefly consider the longer-distance cases of (3) where binding was across two clauses. Due to space constraints we cannot show the full derived trees for these examples, but the key observation can be made from the derivation trees for the English example (3a):

(22)



In this case, the d value for the bound variable MCS is 5. In principle, there is no upper bound on the d value, as further clauses could be interposed between the variable and its antecedent in both languages. Expressed in terms of delayed locality, the only distinction to be made between Korean and English is that while *caki* is grammatical where d = 3, the English bound variable pronoun is not. In the next section, we use this observation to define a more general binding constraint.

4. Constraining Binding

As stated earlier, the elementary trees for the bound variables in both English and Korean are formed in such a way that they must compose by way of delayed locality. We therefore propose that locality constraints for bound variables can be expressed in terms of the observed d value for composition of a given bound variable. Rather than the syntactic definition, framed around a seemingly arbitrary choice of nodes which intervene between the variable and its antecedent, our constraint makes direct reference only to a property of the bound variable itself. Further, we restrict our constraint to only making reference to a single d-value: the threshold n + 1, in this case, 3. Given that all derivations making use of delayed locality will have a d of at least n + 1, there are only three logical possibilities which need to be considered:

(23) a. $d = n + 1$

b. $d \geq n + 1$

c. $d > n + 1$

The case in (23a) would be that of a variable which must combine via delayed locality, but can only do so at the minimal threshold value. Looking back to the ungrammatical (1a), if we were to cast the English reflexive *herself* as a bound variable, this would be exactly the right constraint: a limitation to binding by a co-argument. (23b) represents the most lenient constraint, allowing any possible delayed local derivation; this would correspond to Korean *caki*, as we have shown there to be no locality constraints on its use. Finally, (23c) corresponds to what we have seen for English bound variable pronouns: there is no upper limit on how far the variable may be from its antecedent, but it must at least be further than the threshold value for delayed locality.

By restricting our locality constraints to the threshold value, these are the only three logical possibilities available, translating to only three possible types of bound variable: one restricted to co-arguments, one viable for anything but co-arguments, and one which is unconstrained. A variable binding constraint based on the derived syntax tree has no equivalently principled way of keeping down the number of logically possible constraints. Once a binding constraint makes reference to at least one TP/DP node, for example, there is no reason why further counting constraints (at least two, at least three, etc...) could not be formulated. As such, we might expect to find bound variables which must be at least two or three clauses removed from their antecedents. By restricting our locality constraint to the permutations in (23), we reduce the number of possible bound variables in natural language from a theoretically infinite number of possibilities to three, all of which pivot around the threshold value.

All of this is not to say that the derived trees are completely irrelevant. Indeed, we have already assumed a constraint against unbound variables in the derived semantics as a means of blocking certain derivations. So, while we do not make use of the derived syntax tree to formulate our locality constraints, the derived syntax does still play one key role. Aside from locality, English bound variables are subject to another well-known constraint:

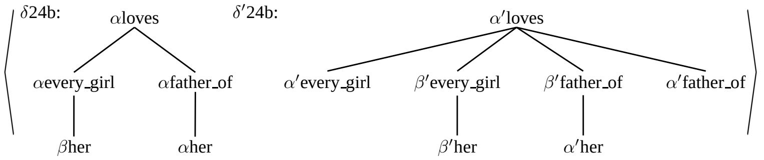
(24) a. * She$_i$ loves every girl$_i$

b. * Her$_i$ father loves every girl$_i$

After quantifier raising, the examples in (24) result in crossover violations, strong and weak, respectively. (24a), the strong crossover case, is already predicted to be ungrammatical under the current analysis, as its derivation tree will be identical to that of (1a), and the d value for the bound variable's delay will be 3.

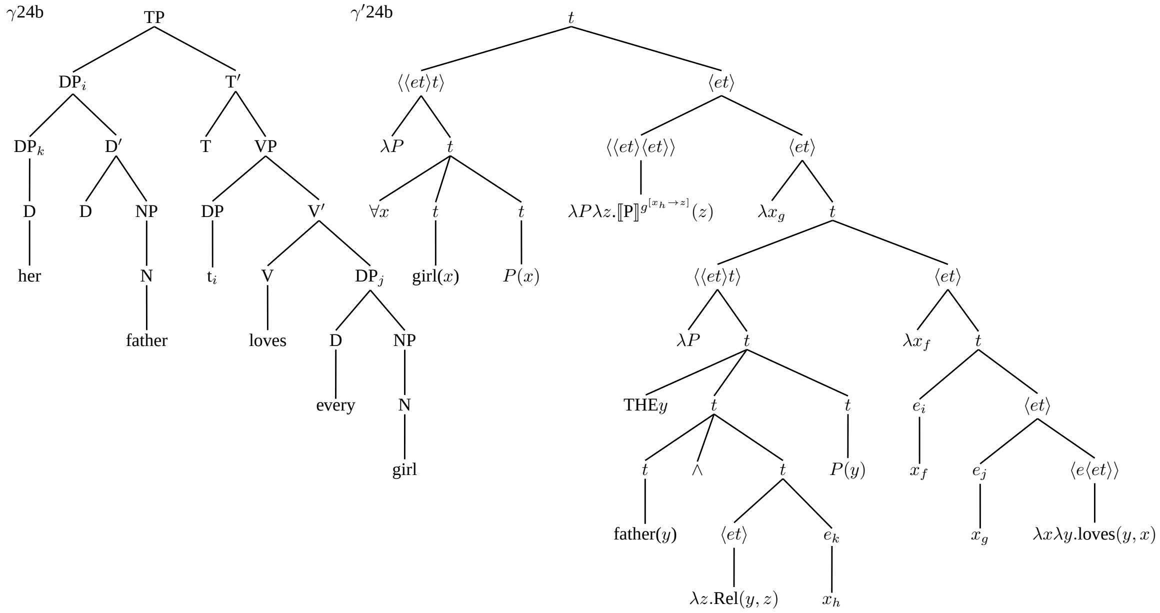
Capturing weak crossover is not so simple though. Recalling the discussion of (1b), there are two possible derivations where there are two GQs, one of which leaves the variable contributed by (α'her) unbound. However, a perfectly legitimate derivation for (24b) is possible, with the derivation trees shown in (25).

(25)



This example cannot be blocked on the basis of the locality constraint, as d=4 for the bound variable MCS. Semantic composition from the derived trees in (26) results in the semantic form in (27) with the variable bound, and the intended meaning intact.

(26)



(27) $\forall x$[girl(x)][THEy[father(y) $\wedge$ Rel(y,x)][loves(y,x)]]

To block this, we impose a c-command constraint between the elementary trees of the bound variable MCS: in the derived syntactic tree, the defective DP* elementary tree must c-command the argument DP tree.[1] In (26), (βher) has adjoined at the root of (αevery_girl), while (αher) substitutes at a higher position in (αloves): the necessary c-command relation does not hold, ruling out this sentence. It is worth noting that this same constraint will rule out the strong crossover violation in (24a) as well. In a sense, the TAG derivation also allows us a more principled distinction between weak and strong crossover, in that the strong crossover case incurs a double violation of both c-command and locality, whereas the weak crossover case only violates the c-command constraint.

[1] Frank (2002) imposes a similar c-command constraint in a proposed MCS analysis of *wh* binding.

5. Conclusion

Having modelled variable binding in STAG, we are thus left with an interacting set of constraints on both the syntactic and semantic sides of the derived structure, as well as on the derivation itself. Because STAG allows us to isolate these specific aspects of a derivation so explicitly, we are better able to form a principled set of constraints than if we just had access to the derived syntactic tree alone. A constraint against unbound variables is most naturally characterized in terms of the derived semantic tree, and a c-command constraint is most naturally defined on the derived syntactic tree. Because these are in essence binary constraints (bound vs. unbound and c-commanded vs. non-c-commanded) these constraints can be formulated in a non-arbitrary manner. However, we argue that the definition of binding locality within a derived syntactic tree cannot be accomplished without defining an arbitrary counting constraint. Having a constraint which makes use of the concept of “at least one” immediately opens up the question of “why not two, three, or four”?

Conversely, the binding locality constraints formulated here, in terms of the STAG derivation’s use of delayed locality works on strictly-defined non-arbitrary terms. The threshold n+1 value is a necessary part of any STAG derivation with delayed locality, definable in terms of the bound variable’s MCS without any reference to any other part of the derivation. We then showed that by restricting our binding locality constraint to the three possible variations around that threshold value, we make the typological prediction that there are only three ways of classifying bound variables in terms of locality. Further, this typological prediction is borne out in that, to the best of our knowledge, all bound variables studied in natural language fall into one of the three categories defined in (23).

The observed pattern with English also opens up the possibility of using the constraints from (23) as a means of defining bound variable systems for languages with multiple types of bound variables. If we were to assume that English reflexives are indeed bound variables, then English could be cast as a system making use of the two complementary classes of bound variable, those defined by (23a) and (23c). Conversely, it has been claimed for Korean that the pronouns *ku* (he) and *kukes* (it) may have bound variable readings as well as *caki*, but in non-local contexts (Kang, 1988). This would then give Korean a system of non-complementary bound variables, making use of (at least) those defined by (23b) and (23c). Thus, in addition to defining a typology of bound variables, the locality constraints defined here can form the basis of a typology of bound variable systems across languages.

In future work with STAG, we hope to identify other phenomena which can be captured using delayed locality. The most natural place to look will be among those cases of long-distance dependency which cannot be reduced to recursive structures. While there is already considerable work showing that *wh*-movement within TAG can be constrained to tree-local dependencies with longer distances being an artifice of successive adjoining operations, there are alternative accounts which treat *wh*-dependencies as multi-component operator-variable structures. While much more work needs to be done, we suspect that there may be promise in the avenue of applying the same derivational locality constraints to certain *wh*-dependencies.

References

Büring, Daniel (2005). *Binding Theory*. Cambridge, UK: Cambridge University Press.

Chiang, David & Tatjana Scheffler (2008). Flexible composition and delayed tree-locality. Gardent, Claire & Anoop Sarkar (eds.), *Proceedings of the 9th International Workshop on Tree Adjoining Grammars and Related Formalisms*, 17–24.

Frank, Robert (2002). *Phrase Structure Composition and Syntactic Dependencies*. Cambridge, MA: MIT Press.

Han, Chung-hye, David Potter & Dennis Ryan Storoshenko (2008). Compositional semantics of coordination using Synchronous Tree Adjoining Grammar. Gardent, Claire & Anoop Sarkar (eds.), *Proceedings of the 9th International Workshop on Tree Adjoining Grammars and Related Formalisms*, 33–41.

Han, Chung-hye, Dennis Ryan Storoshenko & Calen Walshe (in press). An experimental study of the grammatical status of *caki* in Korean. *Proceedings of the 19th Japanese/Korean Linguistics Conference*, University of Hawaii at Manoa.

Joshi, Aravind K., L. Levy & M. Takahashi (1975). Tree adjunct grammars. *Journal of Computing Systems Science* 10:1, 132–163.

Kang, Beom-Mo (1988). Unbounded reflexives. *Linguistics and Philosophy* 11, 415–456.

Kroch, Anthony & Aravind Joshi (1985). The linguistic relevance of Tree Adjoining Grammar. Technical Report MS-CS-85-16, Department of Computer and Information Sciences, University of Pennsylvania.

Nesson, Rebecca & Stuart M. Shieber (2006). Simpler TAG semantics through synchronization. *Proceedings of the 11th Conference on Formal Grammar*.

Nesson, Rebecca & Stuart M. Shieber (2009). Efficiently parsable extensions to Tree-Local Multicomponent TAG. *Proceedings of NAACL 2009*, 92–100.

Shieber, Stuart M. (1994). Restricting the weak generative capacity of Synchronous Tree Adjoining Grammars. *Computational Intelligence* 10:4, 371–385.

Shieber, Stuart M. & Yves Schabes (1990). Synchronous tree adjoining grammars. *Papers Presented to the 13th International Conference on Computational Linguistics*, vol. 3, 253–258.