Deriving syntax-semantics mappings: node linking, type shifting and scope ambiguity

Dennis Ryan Storoshenko
Yale University
Department of Linguistics
370 Temple Street
New Haven, CT 06511
`dennis.storshenko@yale.edu`

Robert Frank
Yale University
Department of Linguistics
370 Temple Street
New Haven, CT 06511
`robert.frank@yale.edu`

Abstract

In this paper, we introduce a type-shifting operation which provides a principled means of describing the derivational links required in Synchronous TAG accounts of quantification. No longer do links appear on root nodes of predicates on an *ad hoc* basis, rather they are generated as a part of a type-shifting mechanism over arguments of the predicate. By introducing to the system a set of temporal variables, we show how this operation can also be used to account for the scope interactions of clausal embedding. We then move on to consider additional cases of multiple clausal embedding and coordination.

1 The Issue

Investigations of the syntax-semantics interface in Tree Adjoining Grammar, particularly those making use of Synchronous TAG, grapple with the limitations imposed by the restrictiveness of tree- or set-local MCTAG. To the degree that they successfully treat the mapping between syntax and semantics in this restricted setting, this provides evidence in favor of Joshi's hypothesis that the mild context-sensitivity of TAG is a fundamental property of grammar. Nonetheless, the analyses that have been put forward are at times ad hoc. One wonders why a certain semantic object is associated with some piece of syntax, and why certain nodes of the syntactic representations are linked to the semantics in one manner as opposed to another. In this paper, we report on our first efforts to formulate principles governing STAG pairings, in an effort to provide a more restrictive framework for characterizing STAG-derivable syntax-semantics mappings.

2 Tree Shapes and Type Shifting

We adopt a traditional view of syntactic elementary trees as the realization of a single lexical predicate and its grammatical "associates" (cf. the Condition on Elementary Tree Minimality and Theta Criterion of Frank (2002)). The corresponding semantic objects are composed from the meaning assignments for the lexical anchor together with the meanings associated with non-projected non-terminals. Substitution nodes are interpreted as typed variables (with types determined by a bijection from syntactic categories to semantic types: DP to type e, NP to type $\langle e,t \rangle$, CP, TP and VP to type t, etc. We follow Pogodalla (2004) in assuming that such variables are bound by (linear) lambda operators, and take syntactic substitution of S into T to correspond to (semantic) function application of T to S. For a syntactic node N targeted for adjoining, we assume that the corresponding node in the semantic representation is embedded beneath an abstracted function variable (with type $\langle \alpha,\alpha \rangle$ where α is the type determined by the category bijection for N). Adjoining of auxiliary tree A to tree T corresponds to application of T to A. We assume that adjoining always applies at nodes to which it may; when no content is added, a semantic identity function is applied. Some linkages between the syntactic and semantic trees are straightforward: non-projected non-terminals are linked to the lambda operators binding their associated variables, while projected nodes in the syntax are linked to lambda operators binding variables of the appropriate $\langle \alpha,\alpha \rangle$ type. This gives rise to a pairing of the sort in Figure 1 for the transitive verb *love*.

What is less clear is how to establish the non-bijective linkages between sites for syntactic at-

Proceedings of the 11th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+11), pages 10–18, Paris, September 2012.

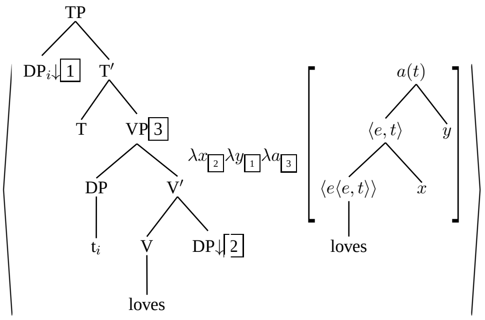

Figure 1: Syntactic and Semantic Tree Pair for *loves*

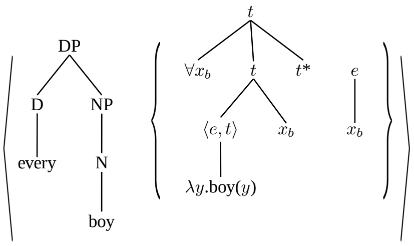

Figure 2: Tree Set for *every boy* (Schema for all Generalized Quantifiers)

tachment and semantic composition. Originating in Shieber and Schabes (1990), and continuing in all of the subsequent STAG-based work on scope we are aware of, it is assumed that the DP position in, say, the subject of a transitive verb-headed elementary tree is linked to both the e-type external argument of the predicate and the t-type root of the tree. This dual-linkage is mirrored in non-STAG accounts of quantification, such as the Hole Semantics-based account in Kallmeyer and Joshi (2003), and subsequent works in that tradition. No matter which type of semantic account the analyst prefers, it is widely accepted that quantification requires this simultaneous access to both an argument position and the root of a tree. Derivationally, this is of course simply a matter of tree-local MCS combination, but in STAG, there is the additional wrinkle of derivational links. Such multiple linkages are crucial for the establishment of scope for quantificational DPs, represented as multi-component sets (MCSs) in the semantics, but not the syntax, as in Figure 2. The variable component of this MCS substitutes into the e-type argument slot, while the t-recursive scope auxiliary tree adjoins at the semantic predicate's t root. It is difficult to see what within the verbal predicate itself directly motivates a link between the DP syntactic position and the t adjoining site in the semantics. We will assume that only the linkage between the syntactic position and the semantic argument slot is basic, as given in Figure 1. Once these are established, semantic trees can undergo an operation that creates multiple linkages in a systematic fashion. Specifically, we make use of an operation similar to argument raising from Hendriks (1988). In Hendriks' operation, the type of an argument is lifted (Partee and Rooth, 1983) from its basic e type to the generalized quantifier $\langle\langle e,t\rangle,t\rangle$ type, allowing a raised argument to effectively take scope over the predicate. Application of this operation to the internal argument of a two-place predicate is shown in (1).

(1) $\langle e,\langle e,t\rangle\rangle : f \Rightarrow$
$\langle e,\langle\langle\langle e,t\rangle,t\rangle,t\rangle :$
$\lambda x_e.\lambda T_{\langle\langle e,t\rangle,t\rangle}.T(\lambda y_e.f(y)(x))$

The lambda gymnastics involved here are substantial. We can accomplish a similar effect with the paired STAG structure in Figure 1 in a simpler way, if we allow one of the e-type arguments to be linked to a new functional $\langle\alpha,\alpha\rangle$ variable. We represent this linkage as the combination of two variables under the scope of a single lambda operator, as shown in Figure 3. We take such set-valued lambda operators to encode the fact that the arguments must be introduced in a single derivational step, via combination with a MCS. In order to ensure that the newly introduced functional variable Q does not disturb the surrounding semantic combinations, it is crucial that Q be type-preserving (i.e., of type $\langle\alpha,\alpha\rangle$ for some α).

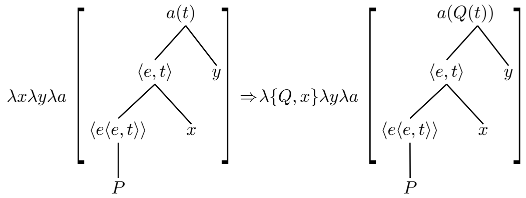

Figure 3: Schematic Example of Type Lifting in Trees (Shown for Internal Argument)

The linkage that has been widely exploited to handle quantifier interpretation fits this pattern: the e-type argument is linked with a $\langle t,t\rangle$ function variable, which will host its scope, shown in Figure 4. Whereas earlier accounts derived quantifier scope ambiguity through underspecified ordering of multiple adjoining at the root t-node of a verbal predicate's semantics, we derive the same ambiguity through underspecified ordering of type

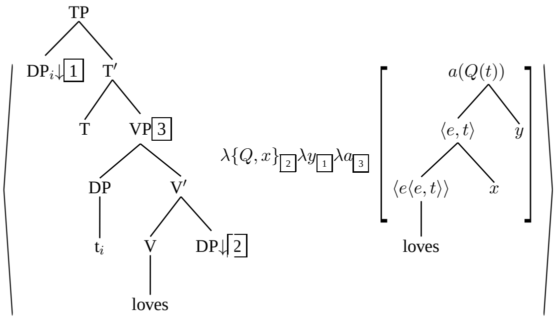

Figure 4: Syntactic and Semantic Tree Pair for *loves* (Type-lifted Internal Argument)

lifting operations, one for each of the predicate's arguments. These iterations of type-shifting take place after the construction of an elementary tree, but before the tree enters into any TAG combinatory operations. That is, the links (and their relative scopes) are all in place before any substitution or adjoining operations take place. What we have gained is that the additional non-bijective link which normally appears by stipulation now has a principled origin in the type-shifting operation which makes it possible for a semantic MCS to combine in a single derivational step.

3 Extending Beyond Quantifiers

Type lifting is not limited to linking e-type variables to quantificational scope. In principle, any argument slot can be linked to an arbitrary type-preserving function, so long as there is a MCS that can satisfy these two positions simultaneously. One case involves infinitival complements to control predicates. Under the analysis of control of Nesson and Shieber (2008), the control predicate's semantic representation is a MCS with an e-type variable to fill the embedded subject argument slot as well as a t-recursive auxiliary bearing the predicate's lexical content. Just as with quantifiers, we link the semantic slot for the e-type subject argument with the root t node, at which the embedding control predicate adjoins. Because this linkage is analogous to the one established in the case of quantifiers, we predict its interaction with other linkages to behave similarly. Specifically, we are led to expect that object quantifiers in the infinitival complement clause should be able to scope out of that clause, past the embedding control predicate (as well as quantifiers in the higher clause). This prediction is correct, as shown in (2).

(2) Someone wants to visit every European city. (want > ∀, ∀ > want)

The example is derivable using the tree set for the control predicate in Figure 6, along with trees for the embedded clause and for the quantifiers, all lexical variants of the trees in Figures 1 and 2. The embedded predicate is shown in Figure 5, with type-shifting having applied in the order which yields surface scope. Recall though that inverse scope is equally possible, as we place no restriction on the order of the applications of type-shifting. The derivation proceeds as in Figure 7, with the order of the two instances of type lifting over the arguments of *to visit* left unspecified.

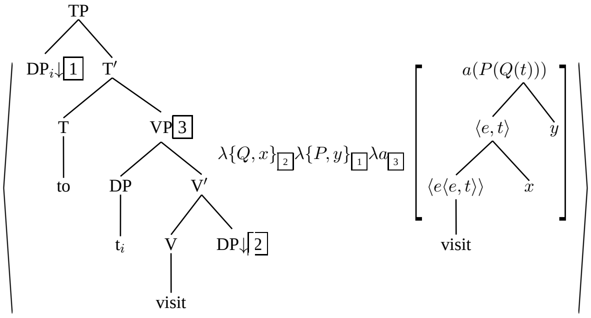

Figure 5: Non-finite predicate *to visit*, type-shifted for surface scope

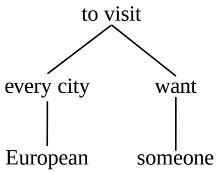

Figure 7: Derivation Tree for (2)

The same scope facts are present in other examples of non-finite clausal embedding, such as raising and ECM, as in (3).

(3) a. Some member seems to like every amendment. (seems > ∀, ∀ > seems)

b. Some member wants every minister to leave. (wants > ∀, ∀ > wants)

Stowell (1982) notes that unlike control, raising and ECM predicates temporally restrict the embedded clause. Matrix predicates routinely specify the embedded clause's temporal interpretation relative to the time of the higher clause, as in

(4) below: depending on the choice of the matrix predicate, the embedded event is understood to take place at a different relative time. For finite clausal complements as in (5), the temporal relation must be conveyed through tense marking in the embedded clause.

(4) a. John regrets missing your talk. (τ(missing-talk) $<$ now)

b. John anticipates missing your talk. (τ(missing-talk) $>$ now)

(5) a. John regrets that he miss**ed** your talk.

b. John anticipates that he **will** miss your talk.

It is straightforward to assume that the dependency in (3) results from the matrix predicate providing a temporal variable to the non-finite embedded clause. We implement this temporal variable using a simplified version of the presentation in Kusumoto (2005); most notably, we omit from our analysis additional situation variables also present in Kusumoto's analysis. This is done purely in the interest of keeping the semantics as clear as possible, and is not intended as an explicit claim that these variables are incompatible with the analysis.

We take the temporal dependency between clauses as in (3) and (4) to indicate multicomponent semantics in the matrix predicate, the use of which must be licensed by type lifting in the embedded clause. Once again, we should expect that this instance of type lifting can be interleaved with those for embedded quantificational arguments, predicting the observed scope facts. We illustrate using the ECM case (3b), beginning in Figure 8 with the elementary trees for the two predicates. ECM *want* is a MCS providing a temporal variable i of type τ, similar to a control predicate providing an argument of type e. The matrix predicate's temporal variable can be saturated by a temporal indexical, whose interpretation varies with the tense of the matrix clause. Crucial to our system is the notion that there is only one such indexical available per derivation. The embedded clause has a similar variable slot, but as just stated, it cannot be similarly filled by an indexical. By type lifting over this position, the ECM predicate may combine in exactly the same way as a quantifier. The derivation in Figure 9 yields the reported scope ambiguity through underspecification of the order of type lifting in the embedded clause. A similar process yields (3a), using the trees in Figure 10, following the derivation in Figure 11.

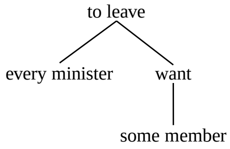

Figure 9: Derivation Tree for (3b)

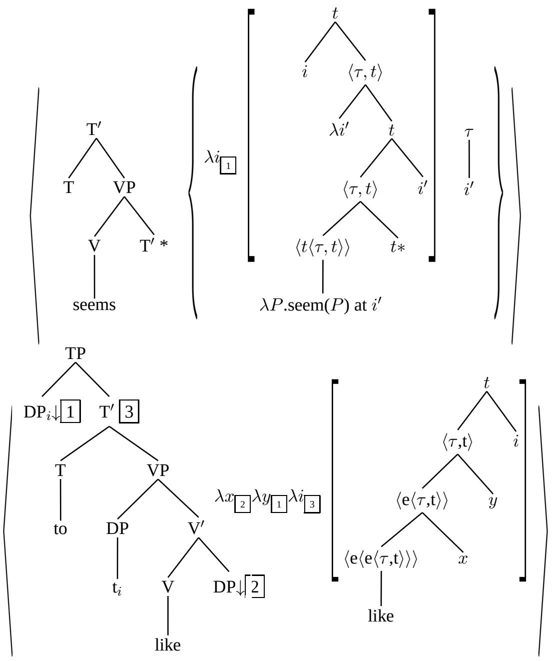

Figure 10: Elementary Trees for (3a)

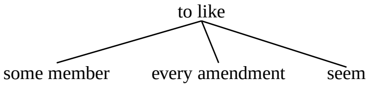

Figure 11: Derivation Tree for (3a)

4 Multiple Predicates

Thus far, we have limited our discussion to simple cases of single clause embedding; in this section we illustrate how the proposed system will interact with multiple embedding, and with coordination.

An example of multiple embedding is given in (6):

(6) John wants Mary to be likely to win.

This is ECM *want*, which will, along with *to win*, use an elementary tree set essentially as in Figure

8. The raising predicate, for the moment, we assume will have the standard TAG syntax of recursion on T′, meaning that it is syntactically possible for both *wants* and *to be likely* to adjoin at different nodes in the *to win* tree. This raises two issues. Firstly, both the ECM and raising predicates would be adjoining at the same node in the semantics, predicting ambiguity between *wants* and *likely*. However, this ambiguity is not found, and only the surface scope of *wants* $>$ *likely* is available. Secondly, there is an issue concerning the interval variables. The embedded predicate *to win* will have one open substitution site for an interval variable in its semantic elementary tree. However, both *likely* and *wants* have such a variable to pass on. It thus seems that under the proposed syntactic analysis, not only do we predict an unobserved ambiguity, but an interval variable will go unused.

To resolve this issue, we propose a tree set for the raising predicate as in Figure 12. Looking first at the semantics, as a clause which will adjoin into a non-finite clause, this passes down an interval variable, as described. However, this predicate itself also requires an interval argument of some sort to saturate its own type τ argument slot, and we assume that only one temporal indexical is available per derivation. If this indexical is to be substituted into the matrix ECM predicate, then that predicate's own interval variable must be the one which substitutes into the raising predicate. That is, *wants* must combine directly with *likely*, not *to win*. This is a welcome finding, as it also predicts the observed scope facts. This then leads us to discuss the syntax of this raising predicate; with an additional degenerate CP node which can serve as the destination of the ECM predicate, direct combination of *wants* into *likely* is now possible, with the CP-recursive *wants* auxiliary tree adjoining to the degenerate CP tree in the set associated with the raising predicate. Type-shifting over the interval variable position in *likely* allows the *wants* MCS to adjoin, and type-shifting over the interval variable in *to win* allows *likely* to adjoin, bringing the ECM predicate along, with both the T′- and CP-recursive adjoinings coming from one elementary tree set.

A slightly different problem arises when combining control with a raising predicate as in the similar (7):

(7) John wants to be likely to win.

First, let us consider the scope facts. As in the previous case, there is only one possible reading here, the surface scope where *want* scopes over *likely*; it is not the case that John is likely to want to win, rather he wants to be likely to win. Thus, the same type of chained derivation would seem to be in order. However, there is an additional complication: we have already made the claim that control predicates pass down a type e argument to the clauses in which they adjoin, not type τ. As a raising predicate, *likely* should never take a type e argument unless an experiencer phrase is added. Further complicating matters is the fact that the type e variable provided by the control predicate is clearly an argument of *win*; this suggests that the derivation which we worked so hard to obviate in the previous case must be the only one available. Both *want* and *likely* should combine directly with the embedded predicate.

However, two new problems present themselves: firstly, given that these predicates will each combine via a type-shift, we predict again there to be a scope ambiguity, contrary to fact. Furthermore, there is the additional question of the open interval variable in the raising predicate. Assuming that *to be likely* here is of the same form as in Figure 12, then what will fill that argument position? We have already claimed that there is only one indexical available, and Stowell's observations make it clear that there is no temporal connection between a control predicate and the clause it embeds. In fact, this second question extends beyond this particular example. The elementary tree for the embedded clause in Figure 5 should likewise require an interval variable which is not provided by the matrix predicate.

To resolve this issue, we propose that non-finite predicates embedded under a control predicate contain a function INF from type $\langle\tau, t\rangle$ to t , given in (8):

(8) INF = $\lambda P.\Diamond\exists i_{Inf}.P(i_{Inf})$

This provides the necessary binding for the interval variable while remaining as non-committal as possible with regards to the actual existence of such an interval. We thus propose a revised version of the non-finite raising predicate as in Figure 13. Only to be used under a control predicate, this gives (local) wide scope to the INF operator. This bears on the second problem, the question of the relative scopes of the two predicates. Earlier, we had stated that because both *want* and *likely* would combine with *win* via a type-shift, their scopes should be permutable, but that only the surface scope is available. We speculate that it is the effect of this INF operator which serves to rule out the reading where *likely* outscopes *want*. This is because such a scope would also give the INF function wide scope over the whole expression, yielding a situation where the widest temporal operator carries this contingent existential, essentially making it possible for there to be no interval at which the described events took place, which is an undesirable result. Nothing in the derivation *per se* blocks this reading, rather a well-formedness constraint on semantic outputs would do so.

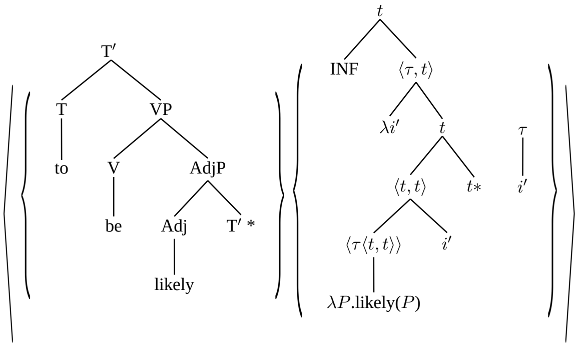

Figure 13: Non Finite Raising Predicate under Control

A reviewer notes that certain cases of Right Node Raising may present a particular challenge for our approach, with an example as in (9):

(9) Every boy supported and every girl protested some amendment.

Specifically, the concern is in deriving the reading in which the shared argument's existential quantifier has narrow scope relative to the universals, since the right-node-raised existential object would appear to be derivationally higher than the subject quantifiers within the conjoined sentences. In fact, such cases can be treated if we adopt the approach to coordination presented in Sarkar and Joshi (1996), and further developed in Han et al. (2008), where examples like (9) are discussed. Under this approach, we allow elementary trees to contain nodes that are marked as shared arguments. When two predicate trees with nodes targeted for sharing (indicated by a circle) are combined, a node-contraction operation applies, such that the relevant shared nodes combine into a single node that is multiply-dominated in both the derived syntax and semantics trees.[1] In Figure 14, we update the relevant Han et al. elementary trees with our new semantic notation, dispensing for the moment with interval variables.

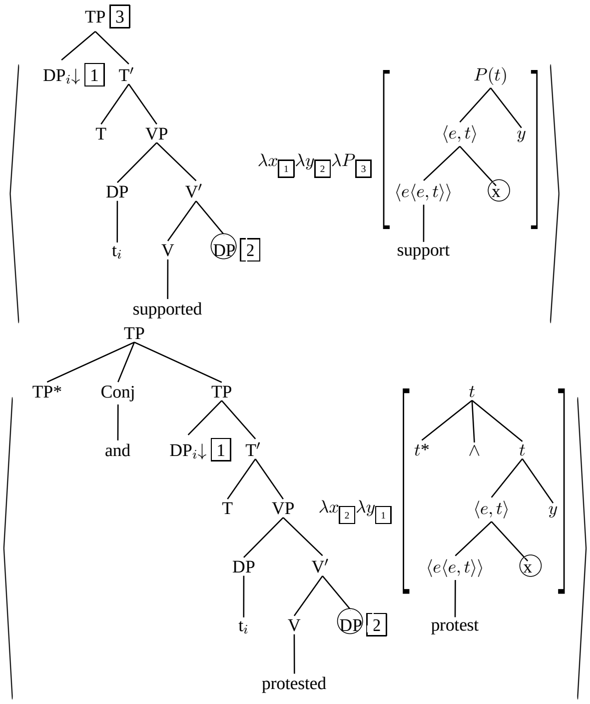

Figure 14: Elementary trees for Coordination

The *support* elementary tree contains three linked nodes: one for the subject, which is a standard type e substitution site, one for the object, which is marked for contraction, and one at the root, which corresponds to λP, which binds a function of type $\langle t, t\rangle$. The coordinator is treated as a functional element in the *protest* tree, projected in accordance with the CETM. The second elementary tree contains two links, one each for the arguments of the verb. These elementary trees however provide two new questions regarding the application of our type-shifting operation. The first concerns the status of the object in the *support* tree. A crucial feature of the Han et al. analysis is that the shared object, *some amendment* in this case, is not duplicated, but rather shared through multiple dominance. As such, its scope part can only combine with one of the two elementary trees. Thus, we do not apply the type-shifting operation at all to the *support* tree, as the only component of the quantificational MCS that is relevant here will be the type e variable. Type

---

[1] See Han et al. for a formal definition of semantic combination with multiple dominance.

shifting the *support* tree would force the quantifier's scope part into a position that does not dominate both of the contracted nodes after *protest* adjoins at the root of *support*, leaving an unbound variable.

The second question concerns the application of type-shifting in the *protest* elementary tree. This is the first time we have explicitly dealt with the question of how to apply the type shifting operation in an elementary tree with more than one available t node. Here, we suppose that the type-shifting operation targets only the root of the elementary tree. That is, both the shared and non-shared arguments of *protest* can take scope over the coordinator. In the case of the shared argument, this is treated as a necessity for proper variable binding in the original Han et al. presentation, an observation which we echo here. However, they treat it as equally necessary for the non-shared argument to take a low scope relative to the coordinator, and here our analyses diverge. We believe it should be possible to derive the reading in which the non-shared argument takes widest scope. Thus, type-shifting will link both arguments to the root of the *protest* tree.

Finally, it is just a matter of determining whether or not there is a possible derivational order which yields the narrow scope of the shared argument. First, we must examine the orderings of type-shifting. By first type-shifting the object position and then the subject position in the *protest* tree, *every girl* will take a wider scope than *some amendment* at the root of that tree. Similarly, the subject position in *support* will also undergo a type-shift, allowing the subject to outscope the coordinating predicate, which does not require a type-shift to combine. Once the predicates are combined, the universals will both outscope the existential, and the coordinator.

5 Type-Shifting and Scope Interleaving

Finally, a reviewer brings the question of whether or not the type-shifting operation will allow for the derivation of quantificational scopes not readily derivable by conventional set-local means. Specifically, an example such as that in (10):

(10) John refused to want to visit every candidate.

The reading of interest here is the one where John refused, for every candidate, to want to visit that candidate. That is, *refuse* $>\forall>$ *want*. Again, because this involves a chain of control predicates, we can dispense with the interval variables for the time being and concentrate on the type e variables. The reviewer is quite correct in suspecting that there is nothing inherent in the type-shifting operation which will allow us to derive this reading. With the tools presently available, *refuse* will combine with *want*, and the embedded quantifier is predicted to scope over or under that compound predicate, yielding either $\forall >$ *refuse* $>$ *want* or *refuse* $>$ *want* $>\forall$.

However, we call readers' attention to Frank and Storoshenko (2012) in this volume. There, we motivate the breaking of predicate elementary trees in the semantics into "scope" and "variable" parts, along the same lines as the generalized quantifiers seen here. Following a suggestion in Nesson and Shieber (2008), a control predicate becomes a 3-member MCS consisting of this scope part containing the substitution site for the external argument, the predicate part, and the type e variable to be passed down. The type-shifting operation described here is not incompatible with that analysis, though its domain of application will naturally be limited to the scope parts of those elementary tree sets which contain substitution sites for type e (or type τ) variables. Again, our account of unspecified ordering of type-shifting applies only to this component of the larger MCS, rather than to the predicate tree.

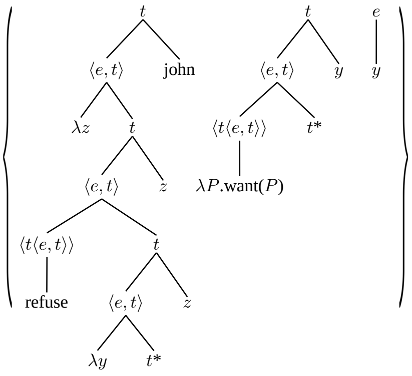

Figure 15: Partial Semantic Derived Tree Set for (10)

To illustrate the example in question, we present the partial derived structure in Figure 15, showing the stage at which *John* has combined with *refuse*, which has in turn combined with *want* after a type-shift over that predicate's open subject position. At this stage, *want* has the described

3-component structure. This will combine with *visit* via type-shifting. One wrinkle at this point is that our type-shifting operation has to this point only dealt with 2-member MCSs. Either the type-shifting operation can be re-defined to allow this triplet, or our preferred interpretation at this point is to maintain that a single link is going to be created, and we rely upon the constraints of the MCS such that the components will eventually collapse into a single tree structure, needing only one link.

As argued in the paper where these structures are introduced, we believe that, with some derivational flexibility, they make certain scopes available which are not otherwise derivable. In this case, as both *want* and *every candidate* will combine via a type-shift targeting the same node (the root of the scope part of *visit*), full freedom for multiple adjoining is possible, so long as all variables are properly bound. Thus, the tree component carrying the universal quantifier should be able to intervene between the two t-recursive components of *want*, deriving the desired reading. For more details on these scope possibilities, we refer readers to the cited paper.

6 Conclusion

In this paper, we have sought a more restrictive framework for linking syntactic and semantic trees in STAG. Combining our modified semantic representations with the type lifting operation provides us a principled account for the creation of the derivational links which captured the simultaneous substitution and adjoining of quantificational MCSs. Extending this operation to type-τ temporal variables, we have shown how embedded quantifiers can scope over raising and ECM predicates. It remains for future work to develop a fully fleshed-out account of the temporal system sketched here, including the links between open type-τ variables and multiple possible syntactic positions where different clauses may adjoin.

Acknowledgments

The authors would like to thank the reviewers for their many helpful comments and questions which have improved the quality of this paper, particularly forcing us to reconcile both our submissions to this volume. All errors remain our own. This work is partially-funded by SSHRC Postdoc Fellowship 756-2010-0677 to Storoshenko.

References

Robert Frank and Dennis Ryan Storoshenko. 2012. The shape of elementary trees and scope possibilities in stag. In *Proceedings of the 11th International Workshop on Tree Adjoining Grammars and Related Formalisms*.

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

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

Herman Hendriks. 1988. Type change semantics: The scope of quantification and coordination. In E. Klein and van Benthem J., editors, *Categories, Polymorphism and Unification*, pages 96–119. Centre for Cognitive Science, Edinburgh.

Laura Kallmeyer and Aravind K. Joshi. 2003. Factoring predicate argument and scope semantics: Underspecified semantics with LTAG. *Research on Language and Computation*, 1:3–58.

Kiyomi Kusumoto. 2005. On the quantification over times in natural language. *Natural Language Semantics*, 13:317–357.

Rebecca Nesson and Stuart M. Shieber. 2008. Synchronous vector TAG for syntax and semantics: Control verbs, relative clauses, and inverse linking. In Claire Gardent and Anoop Sarkar, editors, *Proceedings of the 9th International Workshop on Tree Adjoining Grammars and Related Formalisms*.

Barbara Partee and Mats Rooth. 1983. Generalized conjunction and type ambiguity. In Rainer Bäuerle, Christoph Schwaze, and Arnim von Stechow, editors, *Meaning, Use, and Interpretation of Language*, pages 361–383. Berlin: Mouton de Gruyter.

Sylvain Pogodalla. 2004. Computing semantic representation: Toward ACG abstract terms as derivation trees. In *Proceedings of the 7th International Workshop on Tree Adjoining Grammars and Related Formalisms*.

Anoop Sarkar and Aravind Joshi. 1996. Coordination in Tree Adjoining Grammars: formalization and implementation. In *Proceedings of COLING'96*, pages 610–615, Copenhagen.

Stuart M. Shieber and Yves Schabes. 1990. Synchronous tree adjoining grammars. In *Papers Presented to the 13th International Conference on Computational Linguistics*, volume 3, pages 253–258.

Tim Stowell. 1982. The tense of infinitives. *Linguistic Inquiry*, 13(3):561–570.

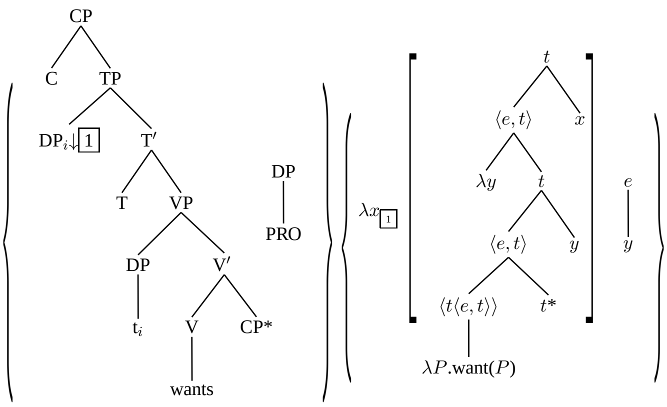

Figure 6: Control Predicate for (2)

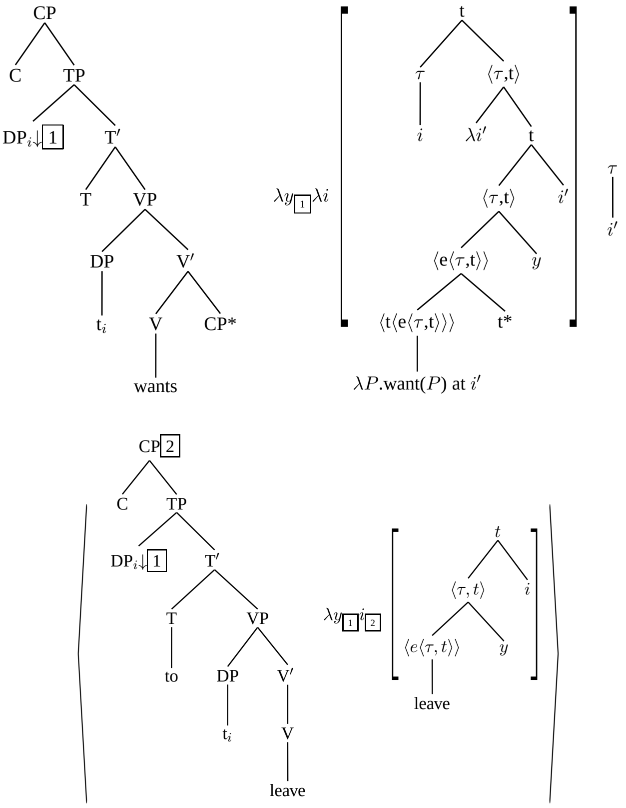

Figure 8: Elementary Trees for (3b)

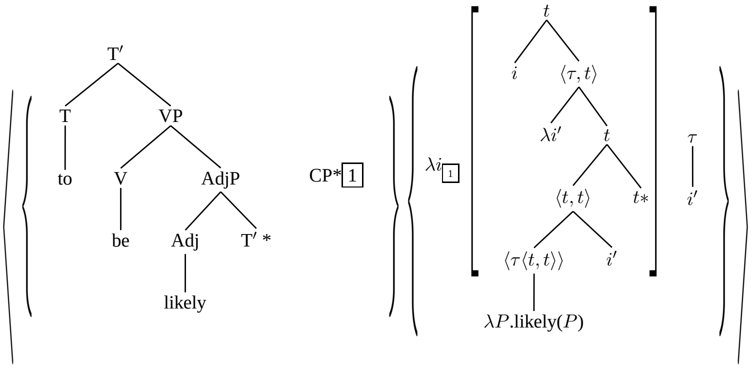

Figure 12: Embedded Raising Predicate for (6)