Synchronous Vector TAG for Syntax and Semantics: Control Verbs, Relative Clauses, and Inverse Linking

Rebecca Nesson
School of Engineering
and Applied Sciences
Harvard University
nesson@seas.harvard.edu

Stuart Shieber
School of Engineering
and Applied Sciences
Harvard University
shieber@seas.harvard.edu

Abstract

Recent work has used the synchronous tree-adjoining grammar (STAG) formalism to demonstrate that many of the cases in which syntactic and semantic derivations appeared to be divergent could be handled elegantly through synchronization. This research has provided syntax and semantics for diverse and complex linguistic phenomena. However, certain hard cases push the STAG formalism to its limits, requiring awkward analyses or leaving no clear solution at all. In this paper a new variant of STAG, synchronous vector TAG (SV-TAG), and demonstrate that it has the potential to handle hard cases such as control verbs, relative clauses, and inverse linking, while maintaining the simplicity of previous STAG syntax-semantics analyses.

1 Introduction

As first described by Shieber and Schabes (1990), *synchronous tree-adjoining grammar* (STAG) can be used to provide a semantics for a TAG syntactic analysis by taking the tree pairs to represent a syntactic analysis synchronized with a semantic analysis. Recent work has used the STAG formalism to demonstrate that many of the cases in which syntactic and semantic derivations appeared to be divergent could be handled elegantly through synchronization. This research has provided syntax and semantics for such diverse and complex linguistic phenomena as relative clauses[1] (Han, 2006; Nesson and Shieber, 2006), nested quantifiers (Nesson and Shieber, 2006), wh-questions (Nesson and Shieber, 2006; Nesson and Shieber, 2007), in-situ wh-questions (Nesson and Shieber, 2007), it-clefts (Han and Hedberg, 2006), and topicalization (Nesson and Shieber, 2007). In these analyses the constraints of the tree-local or set-local MCTAG formalisms have played a critical role in permitting the available semantic readings while ruling out the unavailable ones. This research has demonstrated the value of synchronous grammars for characterizing the syntactic-semantic interface by showing how much more could be done using this simple mechanism than previously thought.

The analysis of nested quantifiers presented by Nesson and Shieber (2006) exemplifies this. Consider the sentence:

(1) Two politicians courted every person at some fundraiser.

We use the synchronous set-local MCTAG grammar in Figure 1 to analyze sentence (1).[2] We depart from traditional TAG notation by labeling adjunction sites explicitly with boxed numbers. The node labels we use in the semantics indicate the semantic types of the phrases they dominate.

Although a nested quantifier may take scope over the quantifier within which it is nested (so-called “inverse linking”) not all permutations of scope orderings of the quantifiers are available (Joshi et al., 2003). In particular, the $every > two > some$ reading is ill-formed (Hobbs and

[1]Both published analyses fail to predict all available scope readings for some sentences. This paper presents a relative clause analysis that addresses this shortcoming.

[2]An alternative analysis exists in which the prepositional phrase modifies the main verb. This derivation is still available and is distinct from the problem case that appears in the literature and that we discuss here.

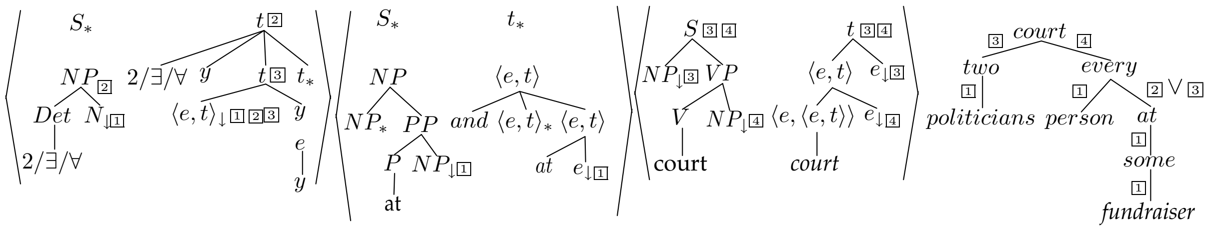

Figure 1: Grammar and derivation for sentence (1): "Two politicians courted every person at some fundraiser." Note that we make use of a higher-order conjunction operation here (and elsewhere), which conjoins properties "pointwise" in the obvious way.

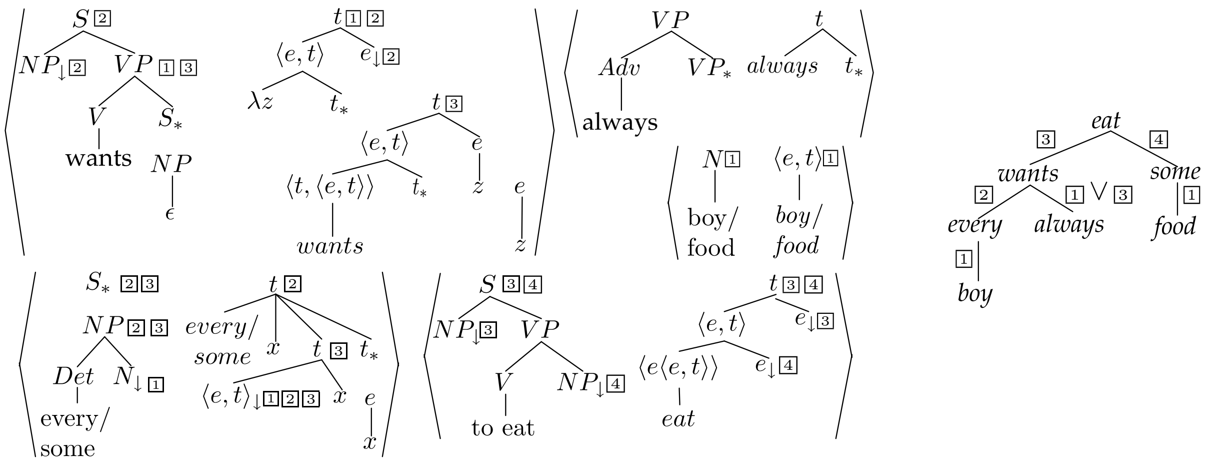

Figure 2: Synchronous TL-MCTAG grammar and derivation for sentence (2): "Every boy always wants to eat some food."

Shieber, 1987), and the *some* > *two* > *every* reading, while formally expressible has been claimed to be not semantically available (Fodor, 1982; Horn, 1974).[3] In our analysis, because the nested quantifier is introduced through the prepositional phrase, which in turn modifies the noun phrase containing the nesting quantifier, the two quantifiers already form a set that operates as a unit with respect to the rest of the derivation. Without any further stipulation, all and only the attested four readings are generated.

However, the simplicity and constrained nature of the STAG approach brings with it serious challenges of expressivity. Certain hard cases push the STAG formalism to its limits, requiring awkward analyses or leaving no clear solution at all.

In this paper we define a new variant of STAG, synchronous vector TAG (SV-TAG), and demonstrate that it has the potential to handle hard cases such as control verbs, relative clauses, and inverse linking, while maintaining the simplicity of previous STAG syntax-semantics analyses.

[3]But see the study by VanLehn (1978) for a contrary view on which this reading is merely dispreferred. We return to this issue later.

2 Difficult Cases for STAG Syntax and Semantics

The elegance of the STAG analysis is encouraging. However, certain cases seem to require more flexibility than the previous analysis, couched in tree- and set-local MCTAG, provides. For instance, as mentioned above, some accounts (VanLehn, 1978; Hobbs and Shieber, 1987) indicate that a fifth scope reading is possible in sentences like sentence (1). We illustrate the limitations of STAG with two further examples involving the semantics of control verbs and relative clauses.

2.1 Control Verbs

Consider the sentence:

(2) Every boy always wants to eat some food.

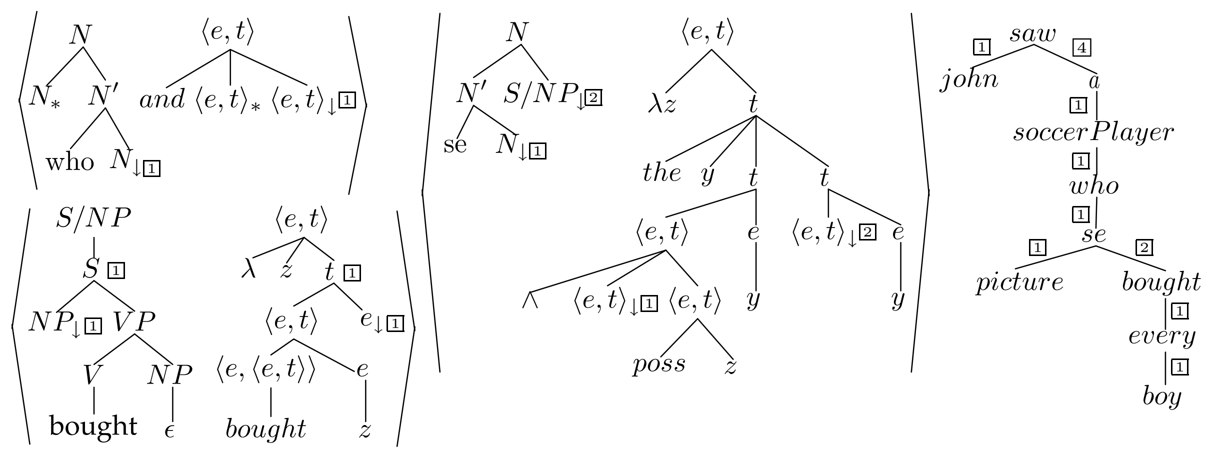

Figure 3: Additional grammar and derivation for sentence (3): "John saw a soccer player whose picture every boy bought." The tree sets for nouns, quantifiers, and the verb *saw* have the same structure as those in Figure 2.

With appropriate context, sentence (2) can produce the scope ordering $always > some > every > wants$.[4] However, a straightforward STAG analysis of the sentence produces a derivation that is incompatible with this reading. Both the derivation of the sentence and the elementary trees for *wants* and *always* are given in Figure 2. If *always* adjoins at link $\boxed{1}$ and *every* adjoins at link $\boxed{3}$ they become indivisibly attached to each other and *some* cannot intervene between them. If *always* adjoins at link $\boxed{3}$ instead, the scope reading $every > some > always > wants$ will be produced. But there is no way to generate the reading $always > some > every > wants$. In order to produce this reading the scope of *every* and the scope of *always* must be prevented from becoming attached to each other before they multiply adjoin with *some* at the root of *eat*.

2.2 Relative Clauses

Consider the sentence:

(3) John saw a soccer player whose picture every boy bought.

In this sentence *every* can outscope the implicit quantifier in *whose*, giving the reading where each boy bought a different picture of the soccer player.[5] However, as shown in Figure 3, because *every* adjoins to *bought* and *bought* substitutes into *whose* below the scope of *whose*, there is no way for the scope of *every* to attach above *whose*. As with the earlier problems, what is required is the ability to delay the attachment scope of *every* to allow it to attach higher in the derived tree.

These examples demonstrate that STAG requires further development to be able to express the full range of readings that quantificational phenomena generate.

3 Synchronous Vector-TAG

A simple solution to this problem would merely relax the set-locality of the semantic MCTAG in the presented grammar. However, this introduces at least two problems. First, the complexity of non-local MCTAG is prohibitive. Second, by eliminating set-locality, the readings generated become extremely hard to control. To remedy these problems, we propose the use of vector TAG (Rambow, 1994), a computationally more tractable and expressively more controllable multi-component TAG formalism as the base formalism to synchronize.

A Vector-TAG (V-TAG) is a 5-tuple (N, T, S, V) where N and T are disjoint sets of nonterminal and terminal symbols, respectively; $S \in N$ is the start symbol; and V is a set of sets of trees, called vectors.[6] The vectors in V

[4]The problem arising from sentence 2 was pointed out to us by Maribel Romero and students at the University of Pennsylvania.

[5]The problematic characteristics of this example were pointed out to us by Chung-hye Han.

[6]In Rambow's original definition the sets in V were partitioned into two sets, V_I and V_A, where the sets in V_I were constrained to include at most one initial tree, and the sets in

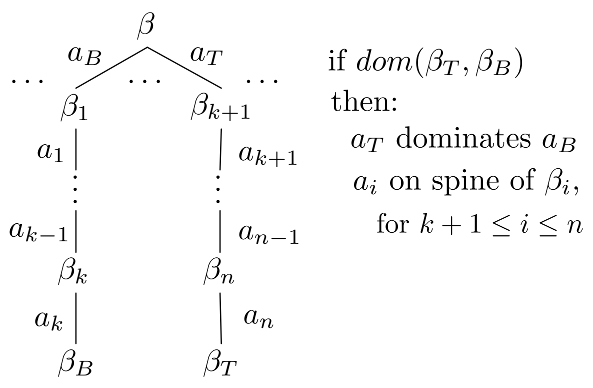

Figure 4: Schematic diagram of a V-TAG derivation tree.

possess dominance links. For a vector $\tau \in V$ the dominance links form a binary relation *dom* over the set of nodes in the trees of τ such that if $dom(\eta_1, \eta_2)$, then η_1 is the foot node of an auxiliary tree in τ, and η_2 is any node in any tree of τ. A *strict* V-TAG is one in which all trees in a vector are connected to each other via dominance links. We use an even stronger constraint in the analyses presented here in which the dominance links in a vector must form a total order over the trees. We call the unique tree in the vector that does not dominate any other tree the *foundation tree*. We distinguish individual trees in a vector with subscripts numbered from 0 starting with the foundation tree.

A derivation in a V-TAG is defined as in TAG. There is no locality requirement or other restriction on adjunction except that if one tree from a vector is used in a derivation, all trees from that vector must be used in the derivation.[7] In addition, all adjunctions must respect the dominance relations in that a node η_1 that dominates a node η_2 must appear on the path from η_2 to the root of the derived tree. If a tree with foot η_1 multiply adjoins at the same location as a tree containing a node η_2 that is dominated by η_1, the tree containing with η_1 must appear higher in the derived tree. Rambow (1994) defines integrity constraints for V-TAG that limit the locations where trees in a vector may adjoin. An *integrity constraint* placed on a node in an elementary tree dictates that the node may not be on the path between two nodes connected by a dominance link.

The derivation tree for a V-TAG may be constructed just as for an MCTAG or STAG where the nodes of the tree are the tree sets and the branches of the tree are labeled with the links at which the synchronized operations take place or the address of the adjunction in the case of a non-foundation tree. The base derivation tree can also be elaborated to give a clearer picture of the relationships between individual trees. In an elaborated derivation tree each tree in a vector is represented explicitly and subscripted to indicate its place in the total order of its vector.

In an elaborated derivation tree the non-foundation trees of a vector do not have to be children of the same tree as the foundation tree of their vector. However, the dominance constraints of the vectors must be respected. Well-formedness can be checked on an elaborated derivation tree by finding the nearest common ancestor of any two trees connected by a dominance link and checking that the address on the branch leading to the dominating tree dominates the address leading to the dominated tree and that each tree along the path from the dominating tree to the common ancestor adjoins along the spine. Figure 4 gives a schematic example of a well-formed elaborated derivation tree.

We define a synchronous V-TAG (SV-TAG) as a set of triples, $\langle v_L, v_R, \frown \rangle$ where v_L and v_R are V-TAG vectors and $\frown$ is a linking relation between nodes in v_L and v_R. A pair (or pair of sets) of trees within each vector are distinguished as the foundation trees. A *foundation adjunction* occurs when the foundation trees drawn from the left and right vectors of $\langle v_L, v_R, \frown \rangle$ adjoin at linked locations in some other vector $\langle v'_L, v'_R, \frown' \rangle$. In contrast to tree-local or set-local MCTAG in which every adjunction site must be marked with a link in order for a tree set to adjoin, in SV-TAG only the adjunction sites where the foundation trees adjoin are marked explicitly with links. The remainder of the trees in v_L and v_R are free to adjoin anywhere in the left and right derived trees, respectively, so long as they obey the constraints of their dominance links. Practically, this definition constrains synchronized

V_A contained only auxiliary trees. We relax the requirements of and distinction between these two sets of sets to allow sets of any combination of initial and auxiliary trees including sets with more than one initial tree.

[7]The definition of V-TAG is very similar to that of non-local MCTAG as defined by Weir (1988) except that in non-local MCTAG all trees from a tree set are required to adjoin simultaneously.

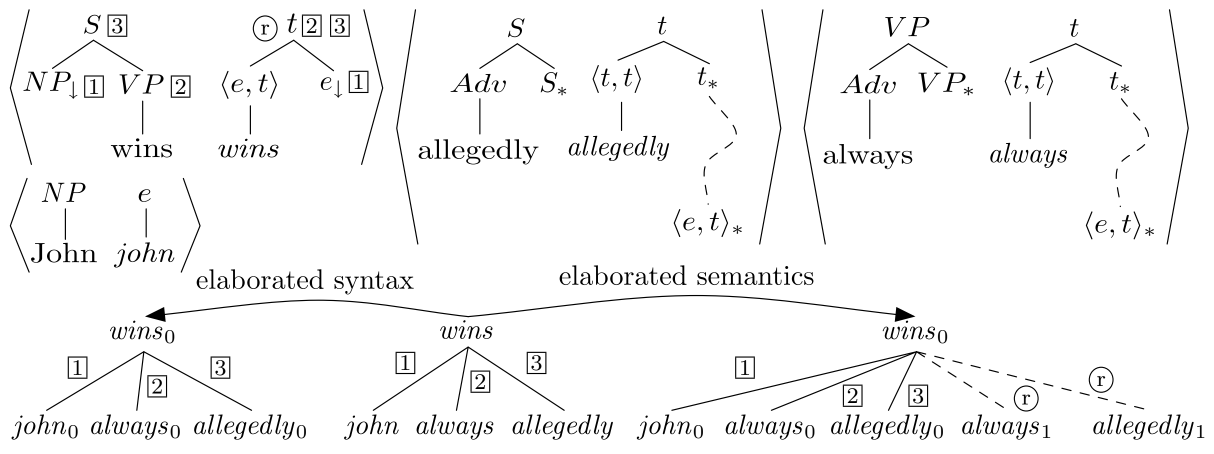

Figure 5: An grammar and derivation trees for sentence (4): "Allegedly John always wins." In the elaborated derivation trees the non-foundation trees are connected with a dashed line. The circled link is simply a shorthand for an address in the tree, not true links in the grammar.

V-TAG vectors to have one synchronized operation with the remainder of the trees adjoining with the usual unconstrained non-locality of V-TAG.

An SV-TAG can be characterized as *simple*, *tree-local*, *set-local* or *non-local* depending on the number and orientation of the link locations in the grammar. If each link has only one location in the left and right grammars then the SV-TAG is called *simple* because the foundation adjunctions on each side of the grammar follow the constraints of a TAG. If links have multiple locations that occur all within one tree on each side of the grammar then the SV-TAG may be termed *tree-local*. When links occur in multiple trees within a vector the SV-TAG is called *set-local* and if link locations of a single link occur in multiple vectors then the SV-TAG is called *non-local*. Although it is possible for foundation trees to occur anywhere in the total order over the trees of a vector, in this analysis we consider only grammars in which the foundation trees do not dominate any other trees in their vector.

Unlike set-local and tree-local MCTAG which are known to be NP-hard to parse (Søgaard et al., 2007), lexicalized V-TAG can be parsed in polynomial time (Rambow, 1994; Kallmeyer, 2007). Although SV-TAG recognition is also NP-hard due to the complexity introduced by synchronization, related work on synchronous unordered vector grammar with dominance links suggests that for a given *simple* SV-TAG grammar a polynomial time tree-to-forest translation algorithm may exist that permits a parse of the syntax of a sentence to be translated into the forest of corresponding semantic trees (or vice versa) (Rambow and Satta, 1996). As with all synchronous-grammar-based analyses, the derivation tree still provides an underspecified representation for the semantics.

3.1 The Derivation Tree

In the STAG model of syntax and semantics the derivation tree is the interface between the two as well as the means for capturing underspecification in the semantics. An SV-TAG permits greater freedom for divergence between syntax and semantics because rather than requiring all trees in a set to be synchronized, in SV-TAG only the foundation trees are synchronized. As a result, underspecification in the SV-TAG model extends beyond multiple adjunction producing different derived trees from the derivation tree. In SV-TAG the additional underspecification results from the locations at which the non-foundation trees ultimately attach. Although the base derivation tree still serves as the connection between the syntactic and semantic derivations and the interface through which they constrain each other, an elaborated derivation tree can help clarify the available readings on each side of the grammar. An example of a grammar, derivation, and elaborated derivation for the following sentence is given in Figure 5.

(4) Allegedly John always wins.

Sentence (4) permits only one reading in which *allegedly* outscopes *always*. This constraint is de-

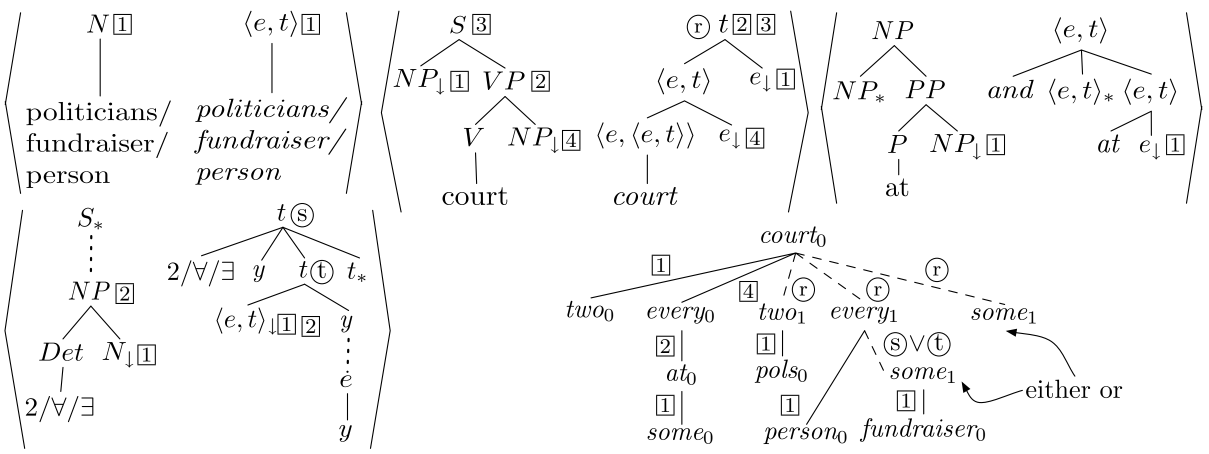

Figure 6: SV-TAG grammar and elaborated semantic derivation for sentence (1): "Two politicians courted every person at some fundraiser." Note that there are three possible locations where *some*$_1$ can adjoin. If it multiply adjoins at ⓡ it must adjoin higher than *every*$_1$ to satisfy its dominance constraint.

termined by the order of attachment of the adverb trees along the VP spine. To enforce this constraint in the semantics we can require that the non-foundation trees of the adverbs attach in the same order as the foundation trees in the shared derivation tree.[8]

4 Applying SV-TAG to the Hard Cases

The additional flexibility provided by SV-TAG permits analysis of the difficult control verb and relative clause examples presented above while still providing a satisfactory analysis of the inverse linking example.

4.1 Inverse Linking

Figure 6 gives a SV-TAG grammar and elaborated semantic derivation tree for sentence (1). The elementary tree sets are similar to the ones presented above except that we have removed the S_* and t_* trees from the elementary tree set for the prepositions *at* and removed all of the non-foundational link locations. The syntactic derivation is straightforward and is presented implicitly in the elaborated derivation tree. The semantic derivation merits closer examination. The tree containing the bound variable of *some* in the semantics foundationally adjoins into *at* at link $\boxed{1}$. The scope tree of *some* is free to adjoin anywhere along the path to the root of the derived tree. It has no site at which to adjoin into the *at* tree, so it must adjoin higher in the derivation tree. The scope tree of *some* may adjoin into either of the two t nodes on the path to the root of the *every* tree while still respecting its dominance link. Adjoining at these two positions will indivisibly connect the scopes of *every* and *some* in both orders as in the STAG analysis of this sentence presented earlier. However, the scope part of *some* does not have to adjoin at these nodes. When *every* foundationally adjoins into *court*, the scope part of *some* will become free to adjoin anywhere between the root of the scope part of *every* and the root of the derivation. Since the only location available for the scope parts of *every*, *two*, and *some* can adjoin are at the root of *court*, this will produce the fifth scope reading in which *two* intervenes between *some* and *every*. The sixth scope reading is prevented by the dominance link requiring the foot of the scope tree of *some* to dominate the bound variable of *some*.

It is interesting to note that the disputed fifth reading requires the scope part of *some* to travel several steps up the derivation tree. Whether there is any relationship between the relative obscurity of this scope reading and the necessity of passing the scope of *some* two levels up the tree may be explored in future work.

4.2 Control Verbs

Figure 7 presents an SV-TAG grammar and the elaborated semantic derivation tree for sentence (2). As with the previous example, the syn-

[8]In the case of two adverbs multiply adjoining at the same location we can require that the order of attachment be consistent across syntax and semantics to produce the same result.

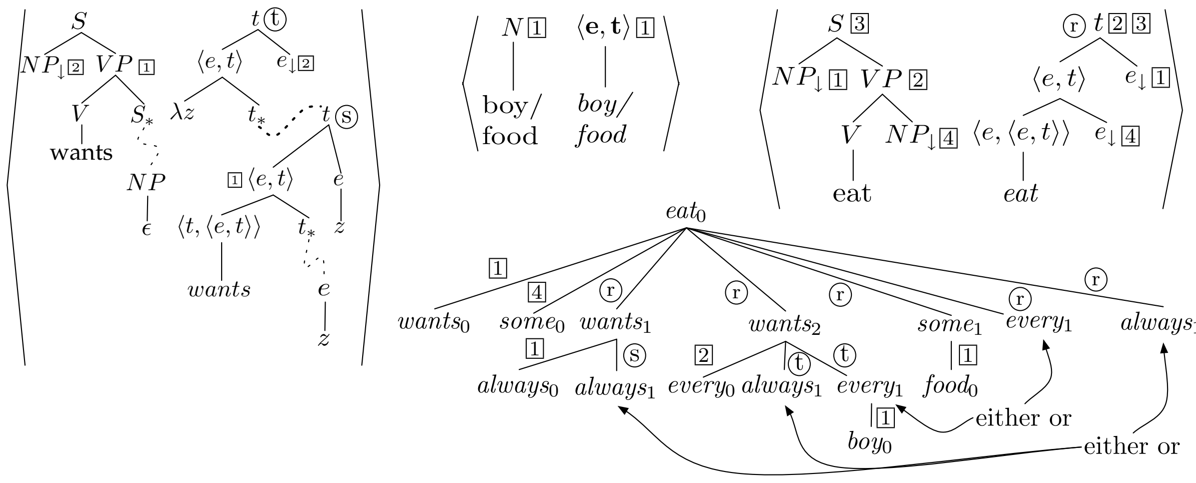

Figure 7: SV-TAG grammar and elaborated semantic derivation tree for sentence (2): "Every boy always wants to eat some food." The tree pair for *always* is as in Figure 5 and the trees for the quantifiers are as in Figure 6.

tactic derivation is straightforward. In the semantics, SV-TAG allows us to produce all six orderings between the quantifiers as well as the de dicto and de re readings of *want*. Both *always* and *every* foundationally adjoin into *wants*. The readings in which *always* and *every* are indivisibly attached to each other as well as the reading in which *some* intervenes between *every* and *always* can be produced by adjoining the dominating trees of *always* and *every* into t nodes of the *wants* tree. The reading in which *some* intervenes between *always* and *every* is produced by the scope parts of *always* and *some* multiply adjoining at the root of *eat*.

Because the scope part of *always* is not part of its foundation it can attach above other scope-takers that attach along the *VP* spine. However, constraints such as the one suggested for sentence (4) may be used to disallow this.

4.3 Relative Clauses

The SV-TAG grammar and derivation tree in Figure 8 achieve the reading that could not be achieved in STAG. Note that the grammar differs only in that the links have been reduced to foundation links. The scope part of *every* is able to pass up through *bought* and is available to adjoin at either of the t nodes in the implicit quantifier in the *se* tree.

Without any constraint the scope part of *every* may continue higher in the derivation to multiply adjoin with the scope of *a* at the root of *saw*. This violates the linguistic generalization that quantifiers may not take scope above a relative clause that contains them. An integrity constraint placed at the root of the *se* semantic tree blocks the scope part of *every* from escaping the relative clause.

5 Conclusion

This paper demonstrates that certain hard cases for synchronous TAG syntax and semantics, such as control verbs, can be successfully analyzed using SV-TAG, a synchronous variant of V-TAG defined herein. SV-TAG maintains the simplicity inherent in the synchronous grammar approach to modeling the syntax-semantics interface, provides the derivation tree as an underspecified representation of the semantics, and is likely to be efficient to process.

References

Fodor, Janet D. 1982. The mental representation of quantifiers. In Peters, S. and E. Saarinen, editors, *Processes, Beliefs, and Questions*, pages 129–164. D. Reidel.

Han, Chung-Hye and Nancy Hedberg. 2006. Pied-piping in relative clauses: Syntax and compositional semantics based on synchronous tree adjoining grammar. In *Proceedings of the 8th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+ 8)*, pages 41–48, Sydney, Australia.

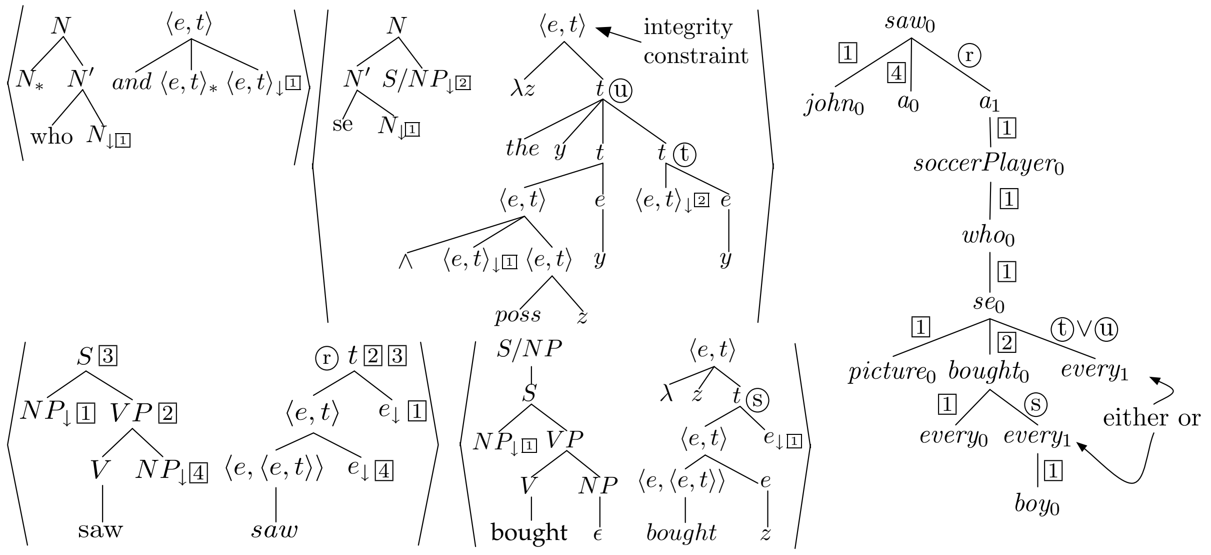

Figure 8: SV-TAG grammar and derivation tree for sentence (3): "John saw a soccer player whose picture every boy bought." Tree sets for nouns and quantifiers are as in earlier figures.

Han, Chung-Hye. 2006. A tree adjoining grammar analysis of the syntax and semantics of it-clefts. In *Proceedings of the 8th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+ 8)*, pages 33–40, Sydney, Australia.

Hobbs, Jerry and Stuart M. Shieber. 1987. An algorithm for generating quantifier scopings. *Computational Linguistics*, 13(1-2):47–63.

Horn, G. M. 1974. *The Noun Phrase Constraint*. Ph.D. thesis, University of Massachusetts, Amherst, Massachusetts.

Joshi, Aravind K., Laura Kallmeyer, and Maribel Romero. 2003. Flexible composition in LTAG: Quantifier scope and inverse linking. In Harry Bunt, Ielka van der Sluis and Roser Morante, editors, *Proceedings of the Fifth International Workshop on Computational Semantics IWCS-5*, pages 179–194, Tilburg, January.

Kallmeyer, Laura. 2007. A declarative characterization of different types of multicomponent tree adjoining grammars. In Georg Rehm, Andreas Witt, Lothar Lemnitzer, editor, *Datenstrukturen für linguistische Ressourcen und ihre Anwendungen - Proceedings der GLDV-Jahrestagung 2007*, pages 111–120. Gunter Narr Verlag, Tuebingen University, April.

Nesson, Rebecca and Stuart M. Shieber. 2006. Simpler TAG semantics through synchronization. In *Proceedings of the 11th Conference on Formal Grammar*, Malaga, Spain, 29–30 July.

Nesson, Rebecca and Stuart M. Shieber. 2007. Extraction phenomena in synchronous tag syntax and semantics. In *Proceedings of the Workshop on Syntax and Structure in Statistical Translation*, Rochester, NY, April.

Rambow, Owen and Giorgio Satta. 1996. Synchronous models of language. In Joshi, Aravind K. and Martha Palmer, editors, *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 116–123, San Francisco. Morgan Kaufmann Publishers.

Rambow, Owen. 1994. Formal and computational aspects of natural language syntax. PhD Thesis, Department of Computer and Information Science, University of Pennsylvania.

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

Søgaard, Anders, Timm Lichte, and Wolfgang Maier. 2007. On the complexity of linguistically motivated extensions of tree-adjoining grammar. In *Recent Advances in Natural Language Processing 2007.*

VanLehn, Kurt. 1978. Determining the scope of English quantifiers. Technical Report 483, MIT Artificial Intelligence Laboratory, Cambridge, MA.

Weir, David. 1988. Characterizing mildly context-sensitive grammar formalisms. PhD Thesis, Department of Computer and Information Science, University of Pennsylvania.