Research on Language and Computation **2:** 281–305, 2004.
© 2004 *Kluwer Academic Publishers. Printed in the Netherlands.*

On Relations of Constituency and Dependency Grammars

MARK DRAS[1], DAVID CHIANG[2] and WILLIAM SCHULER[2]
[1]*Institute for Research in Cognitive Science, University of Pennsylvania, Suite 400A, 3401 Walnut Street, Philadelphia, PA 19104-6228, (E-mail: madras@linc.cis.upenn.edu);* [2]*Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104 (E-mail: {dchiang,schuler}@linc.cis.upenn.edu)*

Abstract. This paper looks at integrating dependency and constituency into a common framework, using the TAG formalism and a different perspective on the meta-level grammar of Dras (1999a) in which the meta level models dependencies and the object level models constituency. This framework gives consistent dependency analyses of raising verbs interacting with bridge verbs, additionally giving a solution to a problem in Synchronous TAG, and gives appropriate analyses of subject-auxiliary inversion. This and other evidence suggests the integration of dependency and constituency is a useful avenue to explore.

Key words: constituency-based grammar, dependency-based grammar, Tree Adjoining Grammar

1. Introduction

1.1. BACKGROUND

English-language linguistics has been dominated since the middle of the last century by grammar formalisms based on constituency. However, dependency-based formalisms also have a long history, arguably longer than constituency grammars. In addition there is a strong body of work in modern times on dependency by Europeans such as Tesnière (1959), Sgall et al. (1986) and Mel'čuk (1988); and there is now something of a resurgence of interest in them in the English-speaking world – in particular, in combining the two in some way to take advantage of both.

Methods for combining dependency and constituency can be categorized into three broad classes. First, there are applications that use dependency (possibly along with constituency) just for practical reasons: for example, the statistical parsing of Magerman (1995), Collins (1997) and others which uses dependency relations between words; the translation methodology of Palmer et al. (1998); the multilingual generation of Iordanskaja et al. (1992); and so on.

Second, there are systems and formalisms which are 'augmented' by selecting aspects of dependency and incorporating them into constituency, and vice versa. This augmentation has a long history as well, before its application to English: the modistic school of grammar of Paris in the late 13th and 14th centuries was a fundamentally dependency-based formalism[1] – and notably the source of the notion of government (Covington, 1984) – but several of its proponents acknowledged that constituency was necessary to handle aspects of language like conjunction, sentence embedding, and impersonal constructions. More recent attempts include the work of Gladkij (1980) on Eastern European languages, reported by Mel'čuk and Pertsov (1987). In the other direction – augmenting mostly constituency-based analyses of English by dependency – Abney (1995) describes a technique of robust parsing, 'chunking', which partially groups words into constituents and then joins these by dependency links; his work covers both a theoretical foundation for this combination based on psycholinguistic and prosodic evidence, and a system for broad-coverage parsing. Other robust parsing systems, such as those based on supertags (Srinivas, 1997), can be viewed as operating similarly. And on the formal side, Hudson's daughter-dependency grammar (Hudson, 1976) adds dependency to constituency; and more generally, the notions of government and of heads of construction have been adapted from dependency grammar into standard Chomskyan analyses (Robinson, 1970).

Finally, there are attempts to integrate constituency and dependency into a single formalism; the difference from the 'augmentations' is that constituency and dependency representations both continue to exist separately but linked in some manner. Early in the recent prominence of constituency-based linguistics, Gaifman (1965) and Hays (1964) expressed dependency grammars using phrase structure rules, and restricted themselves only to projective dependency grammars,[2] showing that these are weakly equivalent to context-free grammars. Then, as part of the Transformational Grammar program, there were proposals to use dependency grammars, rather than context-free or context-sensitive grammars, as the base of a transformational grammar (Robinson, 1970; Vater, 1975), on various grounds including that it allows a neater description of, for example, case phrases than a phrase-structure grammar does (Anderson, 1971), and that it is a weaker theory (Hays, 1964). It was ultimately not adopted because of criticism such as that given in, for instance, Bauer (1979), where it is noted that it is difficult to determine what should be the ultimate head of the sentence, although the determination of the distinguished symbol in phrase structure grammars, and later headedness, has been the subject of similar discussion. Bauer also noted that, given the result of Peters and Ritchie (1973), where transformations are shown to make Transformational Grammar unrestricted in formal power, the base is actually not significant as it is dominated by the transformations. It is, however, interesting to note that some incarnations of Chomskyan theory nonetheless have a 'base' D-Structure component which has properties quite similar to those of dependency grammar: e.g. "We have been tacitly assuming throughout that D-structure is a 'pure' representation of theta structure, where all and only the θ-positions are filled by arguments" (Chomsky, 1986); c.f. the labeled dependency structures of Mel'čuk (1988), which are structures representing headedness and arguments. In all of these, it is possible to view the formalism as consisting of multiple layers, of which one is constituent-oriented, and another is dependent-oriented.

In this paper we explore an integration of formalisms into a common framework in the spirit of this last type of melding of constituency and dependency grammars.

Tree Adjoining Grammar (TAG) is a good candidate for such a framework. Although it does not intrinsically say anything about dependency, TAG assigns *derivation trees* to sentences which are commonly interpreted as dependency structures (Rambow and Joshi, 1997). However, many cases have been pointed out for which TAG derivation trees do not successfully capture linguistic dependencies (Becker et al. 1991; Schabes and Shieber, 1994; Rambow et al. 1995). Schabes and Shieber (1994) revised the standard notion of TAG derivation to better match derivation trees to dependencies, but mismatches still remain.

A related line of research in finding such a framework has been in formalisms which are more powerful than TAG, like set-local multicomponent TAG (Weir, 1988), V-TAG (Rambow, 1994), and most recently, D-tree Substitution Grammar (Rambow et al., 1995). These have been quite successful in modeling the problematic phenomena. However, the philosophy behind TAG and similar formalisms, and the one behind our work in this paper, is that language should be modeled by as tightly-constraining a formalism as possible. The formalisms just mentioned all have formal power beyond that of TAG, which generally adds to parsing and processing difficulty.

Yet another line of research has focused on squeezing as much strong generative capacity as possible out of weakly TAG-equivalent formalisms, as an alternative way of modeling these problematic phenomena (see, for example, Joshi (2000) on what it means to extract more strong generative capacity out of a formalism without increasing its weak generative capacity): tree-local multicomponent TAG (Weir, 1988), nondirectional composition (Joshi and Vijay-Shanker, 1999), and segmented adjunction (Kulick, 1998). We follow this approach.

1.2. MULTI-LEVEL TAGS

One area where the mismatch of TAG derivations and linguistic dependencies has been notably problematic has been in synchronous TAG as defined in Shieber (1994). The languages generated by S-TAGs are of pairs of strings; the formalism can thus represent translation, syntax–semantics mapping, and so on. Under the definition of Shieber (1994) the pairings of strings are induced by isomorphisms between derivation trees. Because of the isomorphism requirement, and TAG's inability to describe certain dependencies, synchronous TAG's ability to define pairings of strings is also limited. These limitations are serious in practice, both for translation (Shieber, 1994) and paraphrase (Dras, 1999b). This was the motivation for (Dras, 1999a) to show that these difficulties could be resolved by the use of a *meta-level grammar*.

A TAG is generally thought of as a set of elementary trees which combine by substitution and adjunction to form a *derived tree*. The process of combining the elementary trees together is recorded (modulo order of application of rewrite steps) in a *derivation tree*.

Let us refine this view somewhat. Weir (1988) showed that the derivation trees of a TAG can be generated by a context-free grammar. We can therefore think of the derivation process as the building up of a context-free derivation tree, followed by the application of a *yield function* f_G, dependent on the grammar G, to produce a derived tree. That is, the derivation tree is a record of the substitutions and adjunctions to be performed, and the yield function actually performs them.

Now, since a TAG yield function maps from trees to trees, nothing prevents us from applying more than one of them. A *k-level TAG* (Weir, 1988) has k yield functions $f_G, f_{G'}, \dots, f_{G^{(k)}}$ (dependent on grammars $G, G', \dots G^{(k)}$) which apply successively to trees generated by a context-free grammar.

In the case of 2-Level TAG, we call G the *meta-level grammar* and G' the *object-level grammar*, because the meta-level grammar generates the derivation trees for the object-level grammar. A *regular form 2-Level TAG* (RF-2LTAG) is a 2-Level TAG whose meta-level grammar is in the regular form of Rogers (1994); this regular form results in a TAG with the weak generative capacity of a CFG, but with greater strong generative capacity (that is, the ability to associate structures with strings). It can be shown that RF-2LTAG is weakly equivalent to TAG (Dras, 1999a).

In synchronous RF-2LTAG (Dras, 1999a), pairings of strings are induced by isomorphisms between meta-level derivation trees. Even though RF-2LTAG is weakly equivalent to TAG, its additional strong generative capacity enables synchronous RF-2LTAG to generate more pairings of strings than synchronous TAG can (Chiang et al., 2000).

In this paper we seek to use this extra strong generative capacity to better describe linguistic dependencies. To do this, we interpret the meta-level derivation trees as dependency structures, instead of attempting to make the object-level derivation trees fit dependencies. In doing this, our approach has similarities to the use of dependency grammars as a base for transformational grammars, in that a dependency representation and a constituency representation are related by TAG yield functions on the one hand and transformations on the other. However, unlike transformational generative grammar our approach is computationally tractable, and, moreover, can be seen as integrating the two representations into a single multidimensional structure, in the sense of Rogers (1997).

We give formal details in Section 2, and then some linguistic applications of this combination of dependency and constituency in Section 3, with further discussion in Section 4.

2. The Formalism

2.1. TAGS

A TAG is a tuple $\langle \Sigma, NT, I, A, F, V, S \rangle$ where Σ is a set of terminal symbols, NT is a set of non-terminal symbols (with $\Sigma \cap NT = \emptyset$), I and A are sets of initial and auxiliary elementary trees, F is a finite set of features, V is a finite set of feature values, and $S \subset NT$ is a set of distinguished symbols, respectively.[3] Elementary trees are trees whose internal nodes are labeled with symbols from NT, and whose leaf nodes are labeled with symbols from both NT and Σ. Auxiliary trees differ from initial trees by additionally having a distinguished leaf node, the *foot node*, which shares the same label as the root.

Elementary trees may be composed into larger trees by the the operations of substitution and adjunction (see Figure 1). Substitution is the attachment of an initial tree (such as that for *John*) at the frontier of another tree, by identifying the root of the initial tree with one of the host tree's leaf nodes (marked with $\downarrow$); the identified nodes must have the same label. Adjunction is the attachment of an auxiliary tree (such as that for *quietly*) into the interior of another tree, by removing the entire subtree at one of the host tree's internal nodes, inserting the auxiliary tree in its place, and re-attaching the removed subtree at the auxiliary tree's foot node (marked with $*$); the root and foot nodes of the auxiliary tree, and the node at which the adjunction takes place, are again required to have the same label. Nodes can be additionally labeled with an NA constraint, which prevents adjunction at that node.

Nodes also have non-recursive top and bottom feature structures which must unify for a complete derivation (see Vijay-Shanker, 1987): each node η has top and bottom feature structures $\eta.t$ and $\eta.b$, with the exception of substitution nodes, which have only top feature structures, $\eta.t$. If a new auxiliary tree with root η_1 and foot η_2 is adjoined at η, then $\eta_1.t$ unifies with $\eta.t$ and $\eta_2.b$ unifies with $\eta.b$; if a new initial tree with root η_3 is substituted at η, then $\eta.t$ unifies with $\eta_3.t$.

In this paper we will use the following definitions to talk about trees and the results of composition:

A *derived tree* is a tree consisting of multiple elementary trees composed together. A *completed derived tree* is a derived tree with no non-terminals at frontier nodes. In a completed derived tree top and bottom feature structures at each node are required to unify.

The *tree set* of a grammar G is the set of all derived trees generated by G; the *completed tree set* of G is the set of all completed derived trees. The *string language* of G is the set of all strings on the frontier of trees in the completed tree set.

Each derived tree also has a corresponding *derivation tree* recording the derivation history, analogous to the derivation tree of a context-free grammar. A node η' of a derivation tree is the child of another node η if the elementary tree corresponding to η' is substituted or adjoined into the elementary tree corresponding to η during the derivation. The derivation tree also records the Gorn address in the

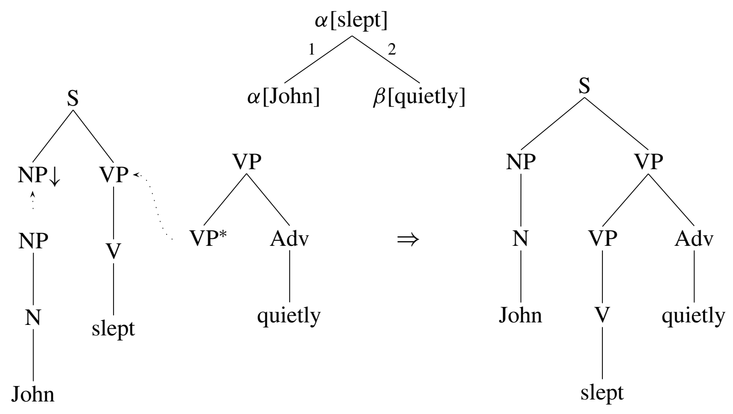

Figure 1. Substitution and adjunction.

elementary tree at which the substitution or adjunction takes place.[4] In Figure 1 the derivation tree is the topmost tree.

The *yield function* of G, $f(G)$, is a way of viewing the process of applying the substitutions and adjunctions specified in the derivation tree in order to produce the tree set of G.

TAGs used for linguistic analysis conventionally take terminal symbols to be words and non-terminal symbols to be part-of-speech or phrasal categories. In this paper we modify this slightly, so that NT contains only a single element, and the part-of-speech or phrasal category is contained in a feature *label*, although graphically we represent this in the traditional way with the node depicted by this label. More detail is in the Appendix; but broadly speaking, the effect is to relax the requirement that nodes that are identified during attachment must have the same part-of-speech or phrasal category label. In making the part-of-speech label part of the node's feature structure, we follow other work in the area (Kasper et al., 95). Diagrammatically, we write a node with top and bottom part-of-speech or phrasal category labels X as X, and a node with top label X and bottom label Y as X/Y.

2.2. 2LTAGS

A 2LTAG is a pair of TAGs $\langle G, G' \rangle = \langle\langle \Sigma, NT, I, A, F, V, S\rangle, \langle \emptyset, NT, I', A', F', \bar{I} \cup \bar{A} \cup \mathbb{N}^*, S'\rangle\rangle$. We call the first member of the pair the *object-level* grammar, and the second member the *meta-level* grammar. Both grammars have the same standard TAG composition operations of substitution and adjunction, although below (and in the Appendix) we note some special features of the use of TAG in this paper.

The meta-level grammar has the following properties:

- the set of 'terminals' is empty;

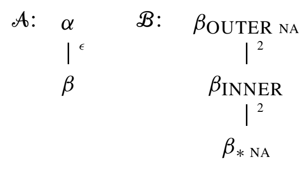

Figure 2. Meta-level grammar for $\{a_1^n \dots a_8^n | n \geq 1\}$.

- the set of 'nonterminals' consists of only a single element (which we will call X);
- the set of feature values consists of the names of the trees of G (in the tuple these are denoted by $\bar{I}$ and $\bar{A}$, the names for the trees in $I, A \subset G$, respectively) and of Gorn addresses in G;
- the set of features, F', has two distinguished elements, *tree* and *addr*;
- *tree* has a ground value in the bottom feature structure of each node;
- *addr* has a ground value in the top feature structure of each node.

Under this definition we can view the yield function $f_{G'}$ as reading the feature values of the nodes in derived trees in G' in order to produce derived trees of G.

Given the above, a node can be represented schematically as:

$$X \begin{matrix} [addr : \eta] \\ [tree : \gamma] \end{matrix}$$

For diagrammatic convenience, however, we will write meta-level elementary trees using the following shorthand:

$$\begin{matrix} |_{\eta} \\ \gamma \end{matrix} \equiv X \begin{matrix} [addr : \eta] \\ [tree : \gamma] \end{matrix}$$

More detailed explanation is in the Appendix.

The result that we want from this definition is that the trees produced by G' look like derivation trees of G. We define the tree set of $\langle G, G' \rangle$, $\mathcal{T}(\langle G, G' \rangle)$, to be $f_G[\mathcal{T}(G')]$, where f_G is the yield function of G and $\mathcal{T}(G')$ is the tree set of G'. Thus, when the elementary trees in the meta-level grammar G' are combined, using the substitution and adjunction operations as defined for TAG, the derived trees can be interpreted as derivations for the object-level grammar G.

We present a simple formal example in Figure 2 (the meta-level grammar) and Figure 3 (the object-level grammar). This grammar generates the string language COUNT-8 $= \{a_1^n \dots a_8^n | n \geq 1\}$. Figure 4 gives a sample derivation for the string $a_1^2 \dots a_8^2$ (with meta-level derivation at the top left, object-level derivation at the top right, and composition of object-level elementary trees at the bottom).

By way of explanation about the notation: we choose to encode the labels of trees in G as feature values in G' because we want 'fundamentally related' nodes to be able to be identified by substitution or adjunction – for example, we want to

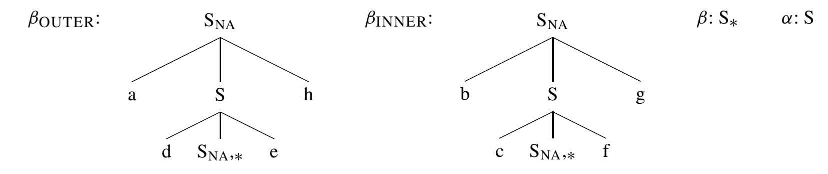

Figure 3. Object-level grammar for $\{a_1^n \dots a_8^n | n \geq 1\}$.

allow the β_{OUTER}-rooted tree $\mathcal{B}$ to adjoin at β in Figure 2 – and if their labels were the strings 'β_{OUTER}' and 'β' this would preclude strict adjunction. A node in the object-level derivation has the tree label in the bottom feature structure because, when another meta-level tree is inserted, we want that elementary tree label to end up below the newly inserted tree; addresses are in the top feature structure because the new object-level trees are inserted at the same address as the tree that has just been 'displaced' by the adjunction. This is compatible with seeing the top feature structure as a 'view from above' – the Gorn address is relative to the root of the object-level tree – and the bottom feature structure as a 'view from below' – the tree name is the identity of the auxiliary tree (Vijay-Shanker, 1987).

The choice of Gorn addresses on nodes versus on arcs is a minor notational variant: the original on nodes, from Weir (1988), is more suitable for our purposes definitionally, although in diagrams we have used the notationally more popular arc labeling. The reasoning can be seen particularly clearly with linguistic examples; see Section 3.1.

2.3. RF-2LTAGS

By itself, 2LTAG has greater generative capacity and recognition complexity than TAG – it can be thought of as 'TAG^2', and as seen in Figures 2 and 3 is able to generate the language COUNT-8 = $\{a_1^n \dots a_8^n \mid n \geq 1\}$ (whereas TAG can only generate COUNT-4), and is also able to generate a broader copy language $\{wwww \mid w \in \Sigma^*\}$ (whereas TAG can only generate the copy language $\{ww \mid w \in \Sigma^*\}$). However, if the meta-level derivations are restricted to a regular form (Rogers, 1994), the object-level derivations will be restricted to a recognizable set like ordinary TAG derivations, so the generative capacity and recognition complexity of the formalism will be constrained to that of TAG (Dras, 1999b).

The regular form condition of Rogers (1994) holds for any TAG if the elementary trees of that grammar do not allow internal spine adjunction (either directly or indirectly) in derivation – that is, adjunction on the path from the root to the foot but not at the root or the foot of an elementary tree. Since the only auxiliary meta-level trees used in our linguistic analyses of Section 3 do not have any internal nodes (unlike in our formal analysis of Section 2.2), our linguistic grammars meet this condition.

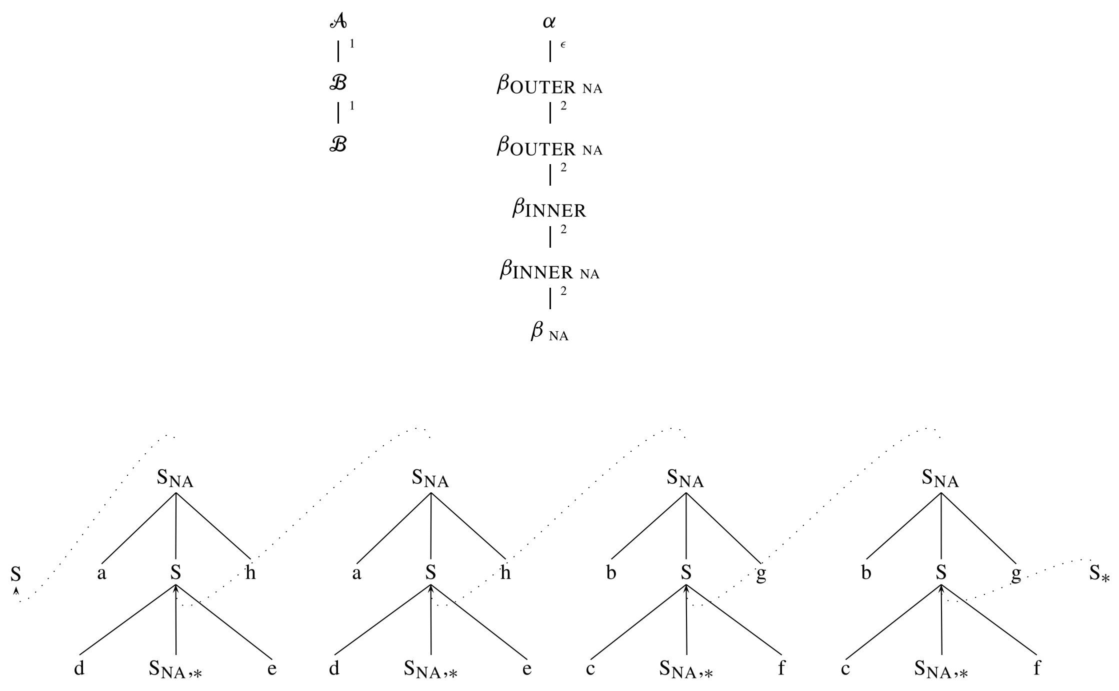

Figure 4. Sample derivation for $a_1^2 \ldots a_8^2$.

2.4. SYNCHRONOUS RF-2LTAGS

Grammars can be paired by some relation, giving rise to a formalism that can be used to model, for example, translation. An early example of this is the formalism of paired context-free rules of Aho and Ullman (1969), which consists of tuples such as the following, where the diacritics indicate a relation pairing non-terminals such that they are rewritten simultaneously:

$$\langle \alpha \rightarrow \beta_1{\boxed{1}}\ \beta_2{\boxed{2}},\ \alpha' \rightarrow \beta_2'{\boxed{2}}\ \ \beta_1{\boxed{1}} \rangle$$
$$\langle \beta_1 \rightarrow \beta_3{\boxed{1}}\ \ \beta_4{\boxed{2}},\ \beta_1' \rightarrow \beta_3'{\boxed{1}}\ \ \beta_4'{\boxed{2}} \rangle$$

In the first pair, for example, when β_1 is rewritten then β_1', sharing the diacritic $\boxed{1}$, must also be rewritten.

A TAG-related formalism similar to this is the Synchronous TAG of Shieber (1994). An early formulation of Synchronous TAG allowed the formalism to generate a language of pairs of strings whose projections were not Tree Adjoining Languages (TALs), even though the component grammars were TAGs: that is, pairing the TAGs increased their formal power (violating what has been termed the *weak language preservation property* by Rambow and Satta (1996)). The revised definition, by pairing the grammars by an isomorphism between derivation trees, leaves the formal power of the grammars unaltered. The S-TAG definition is as follows:

A Synchronous TAG (S-TAG) is a tuple $G = \langle G_L, G_R, \frown \rangle$, where G_L is a TAG, G_R is a TAG, and $\frown$ is a relation specifying linking of addresses in paired elementary trees from G_L and G_R. An S-TAG derivation is a pair $\langle D_L, D_R \rangle$ such that the following hold.

- D_L is a well-formed derivation tree relative to G_L.
- D_R is a well-formed derivation tree relative to G_R.
- D_L and D_R are isomorphic. That is, there is a one-to-one onto mapping f from the nodes of D_L to the nodes of D_R that preserves dominance, i.e. for some nodes $\eta_L \in D_L$, $\eta_R \in D_R$, if $f(\eta_L) = \eta_R$, then $f(\mathit{parent}(\eta_L)) = \mathit{parent}(\eta_R)$, where $\mathit{parent}(\eta_L)$ is the parent node of η_L in D_L (respectively η_R).
- The isomorphic operations are sanctioned by links in the paired elementary trees. That is, if $f(\eta_L) = \eta_R$, then there is a tree pair $\langle \mathit{tree}(\eta_L), \mathit{tree}(\eta_R), \frown \rangle$ in G, where $\mathit{tree}(\eta)$ is the elementary tree corresponding to the derivation tree name η. Furthermore, if $\mathit{addr}(\eta_L) \neq \epsilon$, then there is a tree pair $\langle \mathit{parent}(\eta_L), \mathit{parent}(\eta_R), \frown \rangle$ in G and $\mathit{addr}(\eta_L) \frown \mathit{addr}(\eta_R)$, where $\mathit{addr}(\eta)$ is the Gorn address at which $\mathit{tree}(\eta)$ is substituted or adjoined into $\mathit{tree}(\mathit{parent}(\eta))$.

However, by requiring this isomorphism, S-TAG is limited in the linguistic phenomena it can model; Shieber (1994) notes the problematic case of clitics.[5] Schuler (1999) notes a similar problem in English-Portuguese translation, and attributes it to a mismatch between derivation and dependency structures; we discuss this example in Section 3.2.

We can define the notion of Synchronous RF-2LTAG to resolve this problem. The basic idea is, as before, that dependencies are represented by the highest level of the formalism; then the isomorphism between grammars is established at this level. Formally, the definition is the same as for S-TAG, with the following differences:

A Synchronous RF-2LTAG (S-RF-2LTAG) is a tuple $\langle G_L, G_R, \frown\rangle$, where G_L is an RF-2LTAG, G_R is an RF-2LTAG, and $\frown$ is a relation as before such that if D_L is a well-formed meta-level derivation tree relative to G_L, and D_R is a well-formed meta-level derivation tree relative to G_R, then the conditions between D_L and D_R for S-TAG hold. Under these conditions, Dras (1999b) shows that the weak language preservation property holds, and the projections of the object-level grammar of pairs are still TAGs.

3. Linguistic Analyses Using RF-2LTAG

In this section we present analyses of some linguistic phenomena where TAG has difficulties, and demonstrate how RF-2LTAG, by modeling dependencies, can resolve these problems.

3.1. BRIDGE AND RAISING VERBS

Substitution and adjunction are typically used[6] in linguistic analyses to represent the attachment of arguments and modifiers to the predicates they modify (as in the example in Figure 1), but adjunction can also be used in the other direction, to represent the attachment of bridge and raising predicates to the arguments they predicate over. For example, in an analysis of the sentence,

(1) What does Mary think that John seems to like?

the raising construction *seems* adjoins into the tree for *like* between the subject and the verb; then the bridge construction *Mary thinks* adjoins onto the initial tree on the other side of the subject (see Figure 5).[7] A derivation tree, shown in the top left of the figure, represents the process by which each derived tree was obtained, using nodes for elementary trees and arcs for the substitutions and adjunctions that occurred between them.

One problem with this kind of derivation, first pointed out in Rambow et al. (1995), is that although it comes close to matching the traditional notion of dependency, the derivation for a sentence such as (1) will connect the bridge verb and the lower verb, between which there is no semantic dependency, and will not connect the bridge verb and the raising verb, between which a semantic dependency should exist.

At this point before presenting our analysis it is necessary to clarify the nature of the dependencies we are using. In order to decide which of a pair of words is the governor and which the dependent there are several criteria used (see e.g.

Hudson, 1984), and different conclusions can be drawn. (Compare for example the methodologies of Mel'čuk (1979) and Hudson (1984) in assigning analyses.) We choose the semantic relation between words as our primary criterion; in this our work is similar to Candito and Kahane (1998). Thus *like* is the governor of *seem* and *seem* the governor of *think*, rather than the reverse which would be the case if various syntactic criteria were primary. In any case, the key aspect of dependencies is the pairwise relations between the elements, and the representation in this paper captures these faithfully. Moreover, for applications like translation or statistical modeling, the particular choice of direction is usually immaterial, since the directions are consistently inverted with respect to the syntactic criteria analyses.

In RF-2LTAG we can produce a meta-level derivation tree for (1) which represents the desired dependencies. Given this meta-level derivation, the object-level trees will be somewhat different from those in Figure 5. In part this is because it gives us a neater RF-2LTAG analysis (although it is not necessary for an RF-2LTAG analysis); but additionally, the object-level elementary trees no longer project downward below the lexical predicate of a tree (e.g. *that* in the *think* tree of Figure 5); Frank (2000) argues that this should be the case for TAG elementary trees as a consequence of the Extended Projection Principle.

Now, in our analysis the meta-level auxiliary tree $\mathcal{B}[seem]$ adjoins into the initial tree $\mathcal{A}[like]$ to derive a tree where the node labeled $\beta[seem]$ is between $\alpha[like]$ and $\beta[like]$ (see Figure 6). Viewed as an object-level derivation, this resulting tree has $\beta[seem]$ adjoined at node 2 of $\alpha[like]$, and $\beta[like]$ adjoined at node 2 of $\beta[seem]$. Then $\mathcal{A}[think]$ substitutes into $\mathcal{B}[seem]$ to complete the meta-level derivation, adjoining $\beta[think]$ at the root (address ϵ) of $\beta[seem]$ in the object-level derivation (Figure 7).

A noteworthy characteristic of this analysis, and one that appears in other analyses in this paper, occurs in the $\mathcal{A}[like]$ tree. We split the tree describing the predicate-argument structure for *like* into two parts: the major part $\alpha[like]$ containing the verb, and a separate subject $\beta[like]$; the meta tree is what ties the two together.[8] This $\beta[like]$ can be considered to 'float' – in the derivation level it can move arbitrarily far down, which allows us to group structures at the object level differently. (If it were not there, it would not be possible to have *that seems* grouped together, and the analysis would fail.) Interestingly, this makes subjects look like adjuncts, an idea already discussed in Kayne (1994). It also raises questions about the nature of the lexicalization. We take the trees to be lexicalized at the meta level: for example, $\mathcal{A}$[like] is lexicalized (through α_S[like]) even if its object-level component $\beta_{S/VP}$[like] is not. As a consequence of this, Frank's Condition on Elementary Tree Minimality (CETM) Frank (1992) holds at the meta level rather than at the object level.

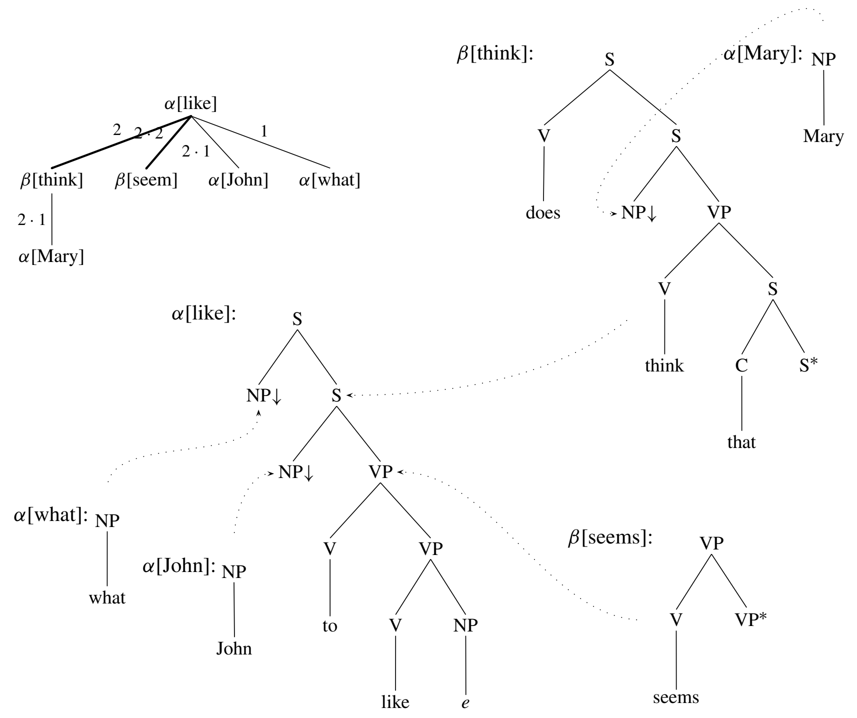

Figure 5. TAG derivation for *What does Mary think that John seems to like?*.

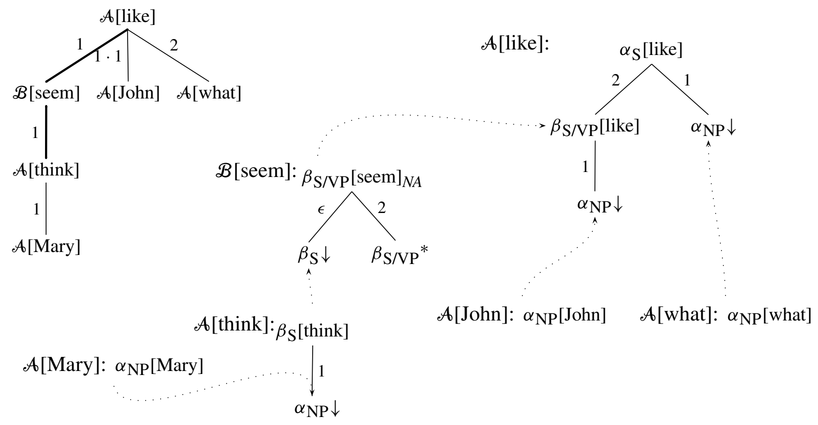

Figure 6. 2LTAG meta-level derivation for *What does Mary think that John seems to like?*.

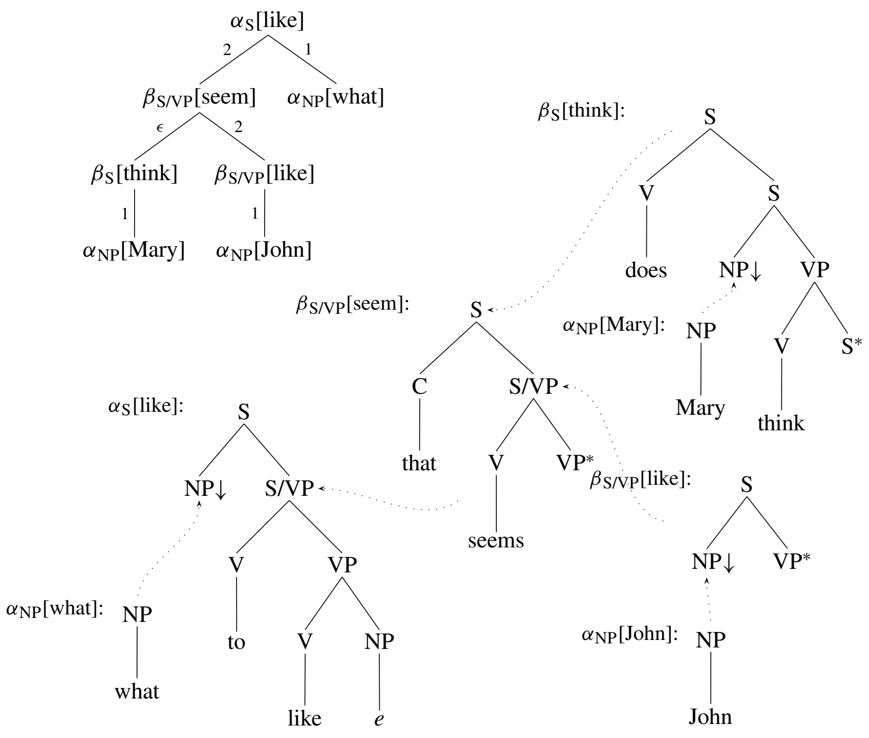
α_S[like]
2
1
$\beta_{S/VP}$[seem]
α_{NP}[what]
ϵ
2
β_S[think]
$\beta_{S/VP}$[like]
1
1
α_{NP}[Mary]
α_{NP}[John]
β_S[think]: S V does S NP↓ VP V think S*
α_{NP}[Mary]: NP Mary
$\beta_{S/VP}$[seem]: S C that S/VP V seems VP*
$\beta_{S/VP}$[like]: S NP↓ VP*
α_{NP}[John]: NP John
α_S[like]: S NP↓ S/VP V to VP V like NP e
α_{NP}[what]: NP what

Figure 7. 2LTAG object-level derivation for *What does Mary think that John seems to eat?*.

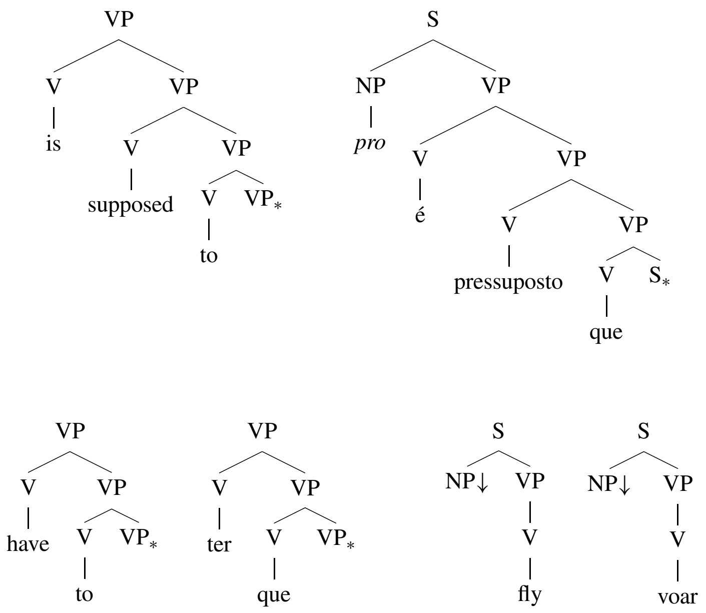
VP V is VP V supposed VP V to VP*
S NP pro VP V é VP V pressuposto VP V que S*
VP V have VP V to VP*
VP V ter VP V que VP*
S NP↓ VP V fly
S NP↓ VP V voar

Figure 8. Elementary trees for sentences (2) and (3).

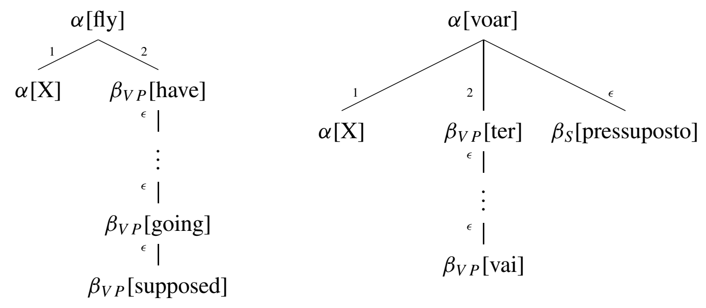

Figure 9. Derivation trees demonstrating *supposed/pressuposto* non-isomorphism.

3.2. TRANSLATION

As mentioned in Section 2.4, there are cases that S-TAG cannot describe because of mismatches in the derivation structure. To solve some of these, Shieber suggests relaxing the requirement that derivations be isomorphic by treating bounded subderivations as elementary, but there are a few cases which remain problematic because they involve ‘unbounded non-isomorphisms’; this is illustrated by a case described by Schuler (1999) and reproduced here. If a predicate is analyzed as a VP-adjunct in one language but an S-adjunct in another, then an unbounded non-isomorphism will arise when this predicate interacts with other VP-adjuncts. Consider the following sentences from English and Portuguese:

(2) X is supposed to (be going to . . .) have to fly.

(3) É pressuposto que X (vai . . .) tem/ter que voar.

We might analyze these sentences (as Schuler does) with the trees in Figure 8, but the resulting derivations for (2) and (3) would be non-isomorphic (see Figure 9).

The unbounded nature of the non-isomorphism can be seen by the nodes β_{VP}[supposed] and β_{S}[presupposto] (which would be paired in a synchronous grammar): in the Portuguese tree β_{S}[presupposto] is immediately dominated by the root α[voar], while in the English tree there is an unbounded number of nodes between β_{VP}[supposed] and the root α[fly].

Schuler (1999) describes a solution to this problem based on a compositional semantics for TAG (Joshi and Vijay-Shanker, 1999) which relies on a mapping of contiguous ranges of scope in source and target derivations. Alternatively, under RF-2LTAG we can analyze the sentences using the trees in Figures 10 to 14; this analysis is the same as in Section 3.1 of bridge and raising verbs. The meta-level derivations in Figure 14 reflect the dependency structure, and are isomorphic. In pairing the grammars in this way we are effectively using dependencies (or at least predicate-argument structures) as an interlingua for translation, and this conveniently satisfies the isomorphism requirement.

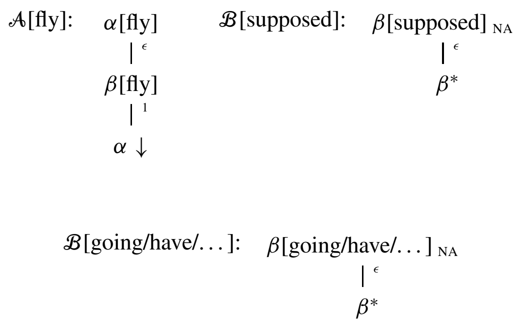

Figure 10. 2LTAG meta-level grammar for sentence (2).

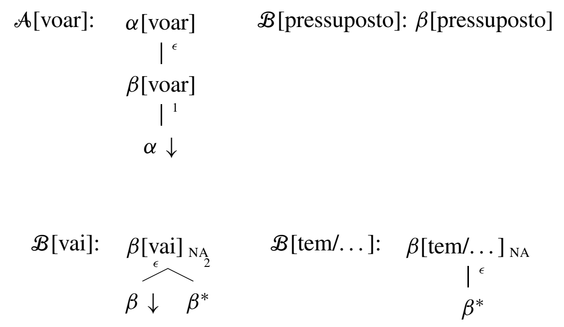

Figure 11. 2LTAG meta-level grammar for sentence (3).

A reason for preferring this alternative is that in Schuler's approach subderivations in the source are not mapped to subderivations in the target, this solution can only be used on individual derivation trees and not (tractably) on entire shared forests of possible derivations (Vijay-Shanker and Weir, 1993). Thus, for example, it is not directly possible to parse a natural language utterance and prune the chart using constraints on a semantic target. In our approach this is not the case.

3.3. RAISING AND SUBJECT-AUX INVERSION

A problem related to example (1) introduced in Section 3.1 occurs with the sentence

(4) Does Gabriel seem to eat gnocchi?

This is problematic for TAG because it is generally assumed, following the CETM of (Frank, 1992), that the functional head *does* must occur in the same elementary tree as the lexical head *seem* with which it is associated. But this is impossible in ordinary TAG because the subject stands in the way. This is exactly parallel to example (1), where we wanted the complementizer *that* to be in the same tree as

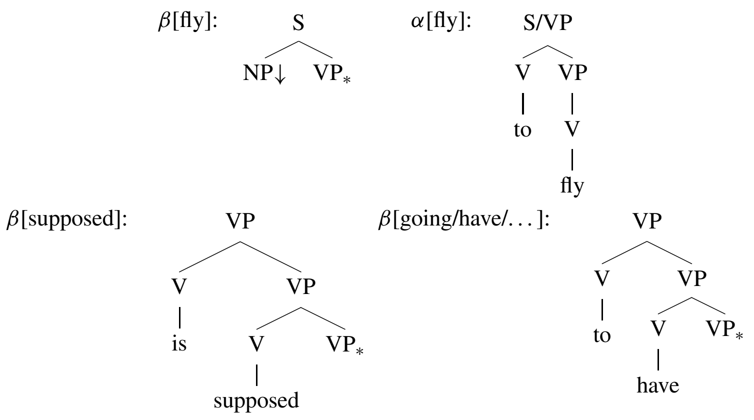

Figure 12. 2LTAG object-level grammar for sentence (3).

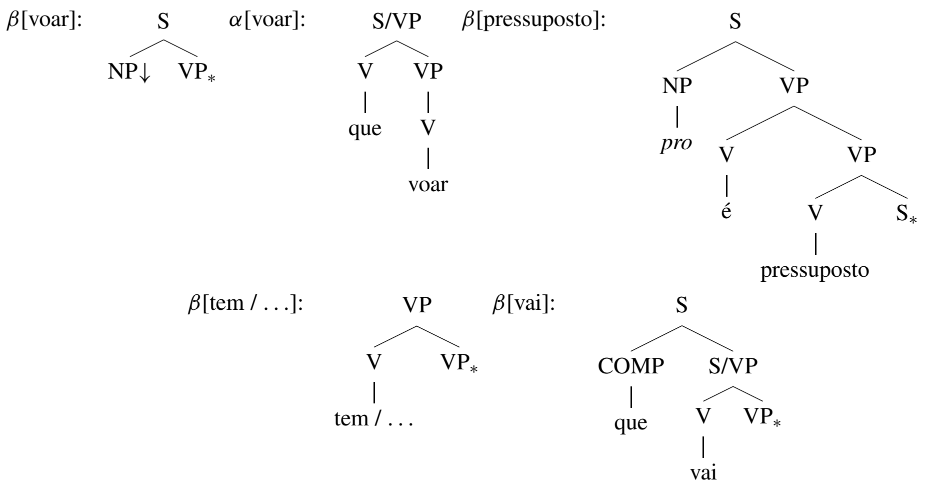

Figure 13. 2LTAG object-level grammar for sentence (3).

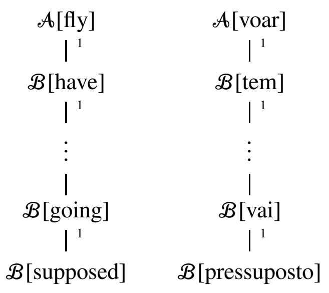

Figure 14. Meta-level derivation figures for (2) and (3).

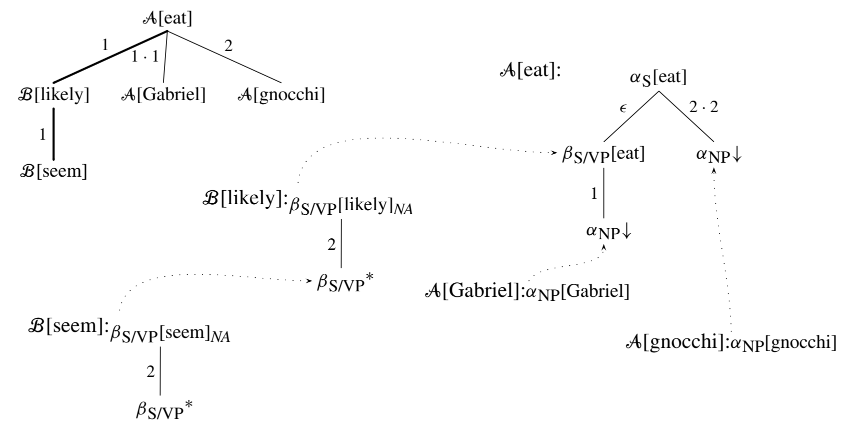

Figure 15. 2LTAG meta-level derivation for *Does Gabriel seem to be likely to eat gnocchi?*.

seems. The solution employed there works here as well: we again simply assume that the CETM applies to meta-level elementary trees instead of object-level elementary trees.

A more familiar solution would be to use tree-local MCTAG (Figure 17), in which a set of trees adjoins simultaneously into a single elementary tree (assuming that the CETM applies to elementary tree sets instead of individual elementary trees). But this solution does not extend to the following:

(5) a. I think that Gabriel seems to be likely to eat gnocchi.

b. Does Gabriel seem to be likely to eat gnocchi?

In both cases tree-locality is violated unless the the tree for *likely* adjoins at the foot of the tree for *seem(s)*, which is normally prohibited. More importantly for present purposes, the derivation would not correctly reflect the dependencies: *eat* would compose with the very highest raising verb.

But in a 2LTAG analysis we again model the dependencies at the meta level, and such sentences are no longer problematic, as shown in Figures 15 and 16.

4. Discussion

4.1. THE OBJECT-LEVEL DERIVATION TREE

It is tempting to think of the object-level derivation tree as an intermediate structure, produced halfway through the transmogrification of the meta-level derivation tree into the object-level derived tree.

But consider the case of context-free grammars. In a CFG derivation, rewrite rules combine to form a derived string, and this process is recorded in a derivation tree. It is reasonable to think of a CFG derivation as the building of a derivation tree, followed by the application of a yield function to produce a derived string. But

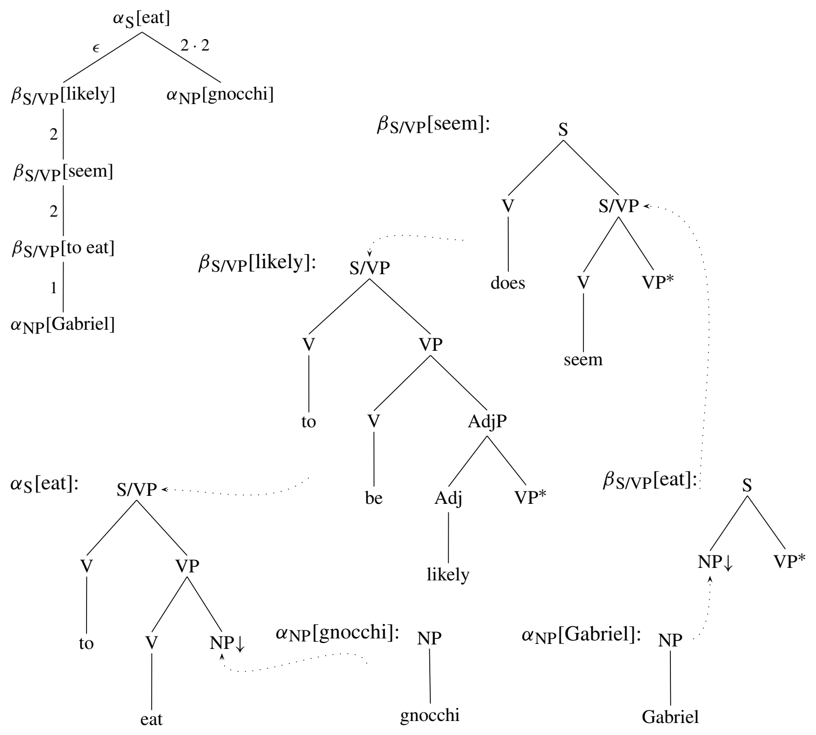
α_S[eat]
ϵ
2 · 2
$\beta_{S/VP}$[likely]
α_{NP}[gnocchi]
2
$\beta_{S/VP}$[seem]
2
$\beta_{S/VP}$[to eat]
1
α_{NP}[Gabriel]
$\beta_{S/VP}$[seem]: S V does S/VP V seem VP*
$\beta_{S/VP}$[likely]: S/VP V to VP V be AdjP Adj likely VP*
α_S[eat]: S/VP V to VP V eat NP↓
$\beta_{S/VP}$[eat]: S NP↓ VP*
α_{NP}[gnocchi]: NP gnocchi
α_{NP}[Gabriel]: NP Gabriel

Figure 16. 2LTAG object-level derivation tree for “Does Gabriel seem to be likely to eat gnocchi?”.

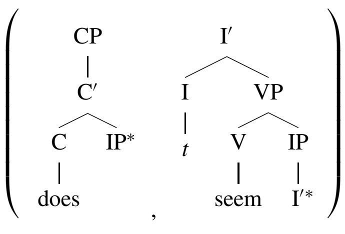

Figure 17. A multicomponent tree set for sentence (4).

it would be strange to conclude that the derivation tree comes prior to the derived string, because the derivation tree contains the derived string on its leaves. All the yield function does is read the leaves off in order; being grammar-independent, it is not an extra step in the derivation but the means of recovering the result of the derivation.

Similarly, Rogers (1997) introduces a notion of TAG derivation trees as three-dimensional trees in which each node is not labeled with the name of an elementary tree, but has a tree of children corresponding to an elementary tree, just as each node in a CFG derivation tree has a string of children corresponding to a rewrite rule. The derived tree (and derived string) are recoverable from these three-dimensional trees by means of a grammar-independent yield function. Thus they integrate derivation trees and derived trees into a single structure. Under the view that TAG derivation trees should represent dependencies, these structures provide an integrated representation of dependency and constituency.

Rogers generalizes this idea further to an infinite hierarchy of multidimensional trees and corresponding formalisms. This hierarchy of formalisms corresponds to Weir's multilevel TAG hierarchy. Thus we can think of a 2LTAG derivation as the building of a four-dimensional structure, followed by successive applications of grammar-independent yield functions to recover the information stored within them. Like the three-dimensional trees produced by TAG, these four-dimensional trees provide an integrated representation of dependency and constituency.

4.2. RELATED APPROACHES

RF-2LTAG follows other work in reconciling dependency and constituency approaches to modeling natural language. One such early integration involved work by Hays (1964) and Gaifman (1965), which showed that projective dependency grammars could be represented by CFGs. However, it is known that there are common phenomena which require non-projective dependency grammars – see Kahane et al. (1998) for further references – so looking only at projective dependency grammars is inadequate. Following the observation of TAG derivations' similarity to dependency relations, other formalisms have also looked at relating dependency and constituency approaches to grammar formalisms.

A more recent, TAG-related instance is D-Tree Substitution Grammars (DSG) (Rambow et al., 1995). In this formalism the derivations are also interpreted as dependency relations, and there is an object-level representation which combines via the operations of subsertion and sister-adjunction. Thought of in the terms of this paper, there is a clear parallel with RF-2LTAG, with a local set representing dependencies having some yield function applied to it, although in the case of DSG it is not explicitly presented this way, and it is not a composition of TAG yield functions. The difference between the two is in the kinds of languages that they are able to describe: DSG is both less and more restrictive than RF-2LTAG. DSG can generate the language COUNT-k for some arbitrary k (that is, $\{a_1^n a_2^n \dots a_k^n \mid n \geq 1\}$), which makes it extremely powerful (by comparison, RF-2LTAG can only generate COUNT-4; and even if the meta-level grammar is not in regular form, 2LTAG can only generate COUNT-8). However, unlike RF-2LTAG it cannot generate the copy language (that is, $\{ww \mid w \in \Sigma^*\}$ with Σ some terminal alphabet); this may be problematic for a formalism modeling natural language, given the key role of the copy language in demonstrating that natural language is not context-free (Shieber, 1985).

Candito and Kahane (1998) propose GAG, an extension of DSG that has a semantic graph as its representation of dependency relations. Its formal properties are as yet unknown, but it seems reasonable to assume that it is more powerful than DSG, given that the derivation graphs of GAG strictly contain the derivation trees of DSG.

Another formalism of particular interest here is the Segmented Adjoining Grammar of Kulick (2000). This generalization of TAG is characterized by an extension of the adjoining operation, motivated by evidence in scrambling, clitic climbing and subject-to-subject raising. Most interestingly, this extension to TAG, proposed on empirical grounds, is defined by a composition operation with constrained non-immediate dominance links that looks quite similar to the behavior of the object level of the formalism described in this paper, which began purely from formal considerations and was then applied to data.

5. Conclusion

From a theoretical perspective, integrating dependency and constituency into a common framework is an interesting exercise. It also, however, proves to be useful in modeling otherwise problematic constructions, such as subject-auxiliary inversion and bridge and raising verb interleaving, one application of which resolves difficulties with the Synchronous TAG formalism. There is much scope for further exploration of the idea, looking at other problematic phenomena, incorporating insights from other formalisms, and investigating more general properties of meta-level grammars.

Acknowledgements

The authors would like to thank the three anonymous reviewers for useful comments, and also to acknowledge the support of grant NSF_STC SBR 8920230.

6. Appendix: Categories as Features

Under TAG as it is typically used, the non-terminals are part-of-speech and phrasal category labels. In our use of TAGs within 2LTAG, we encode the part-of-speech or phrasal category into the node with a feature *label*, and call the symbol for the non-terminal the *signature*. For the purposes of this paper, in an object-level grammar we do not need distinct signatures; additionally, we adopt the convention that root and foot nodes of auxiliary trees have null *label* features in their top and bottom feature structures respectively. Broadly speaking, this says that any tree can potentially be adjoined to any other; what controls whether this is valid is the meta-level grammar. Thus what we show diagrammatically as Figure 18 is actually represented as Figure 19.

In the meta-level grammar, we do not have a *label* feature, but instead have *addr* and *tree* features, which are in the top and bottom feature structures of a node respectively. The *tree* feature indicates which more specific tree of a general signature type is used; the *addr* feature indicates what address the node is attached to in the tree above it. The meta-level

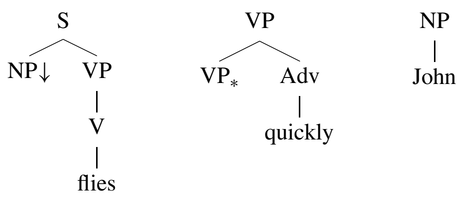

Figure 18. Diagrammatic representation of TAGs.

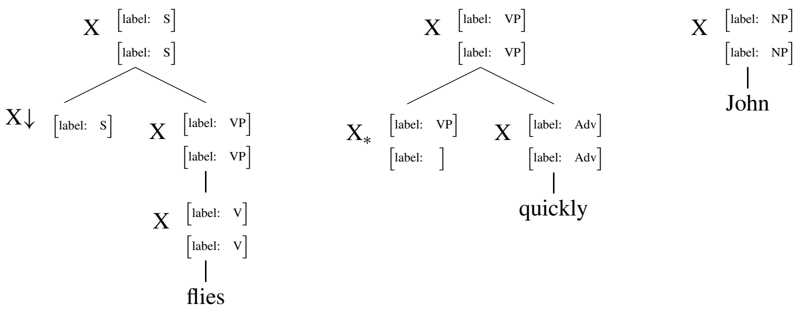
X [label: S] [label: S]
X↓ [label: S]
X [label: VP] [label: VP]
X [label: V] [label: V]
flies
X [label: VP] [label: VP]
X* [label: VP] [label:]
X [label: Adv] [label: Adv]
quickly
X [label: NP] [label: NP]
John

Figure 19. Categories as features.

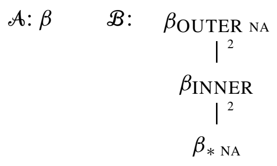

Figure 20. Meta-level grammar for $\{a_1^n \dots a_8^n | n \geq 1\}$.

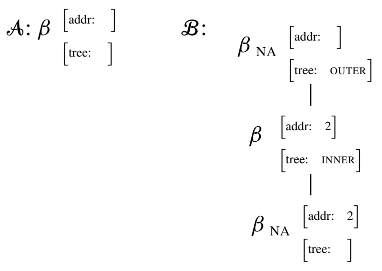

Figure 21. Meta-level grammar for $\{a_1^n \dots a_8^n | n \geq 1\}$.

grammar of Figure 2 is reproduced here in Figure 20, along with the actual feature structure version in Figure 21.

Here, the signature is something that works much like the non-terminals in TAG as it is standardly used, although in Figure 21 we only have β as the signature. In a larger-scale grammar, however, we would typically use tree types for signatures (e.g. βnx0Vnx1, the tree used in XTAG for standard transitive verbs), and the *tree* feature would be the specific instance of that tree (as represented by, say, the lexical item anchoring that tree).

Notes

[1] This use of a dependency formalism was fairly standard before the 20th-century focus on English, and here could be seen as a natural consequence of the fact that the modistic grammarians studied the relatively free word order syntax of Latin: Mel'čuk (1988) comments that constituency-based formalisms only came to be seen as natural ways to describe language because of the English-language focus, and resulting fixed word order bias, of much of modern linguistics.
[2] Roughly, an arc representing a relation between two words x and its dependent y is projective if, for every word w between x and y, w is a dependent (immediately or transitively) of x; a grammar is projective if all of its arcs are projective. More precise definitions are found in the original Lecerf (1960) and the more frequently quoted Robinson (1970). Some natural language phenomena, however, such as English long-distance *wh*-extraction and clitic climbing in Romance, are known to require non-projective analyses.
[3] This is a simplified formulation of the Feature-based TAGs defined in Vijay-Shanker (1987), which are used as standard in the world of TAGs; see e.g. XTAG Research Group (1998).
[4] The Gorn address of a root node is the empty string ϵ; if a node has Gorn address η, then its ith child has Gorn address $\eta \cdot i$, with $i \in \mathbb{N}$ and $\cdot$ as string concatenation (Gorn, 1962).
[5] Contra Shieber (1994), clitics in French are not a problem, although in Spanish they are; see Dras and Bleam (2000).
[6] For example, in the XTAG grammar (XTAG Research Group, 1998). This convention is widely followed.
[7] This analysis of bridge verbs was first proposed by Kroch and Joshi (1987) and is followed in XTAG (XTAG Research Group, 1998). It is not the only possible analysis, but best captures the recursive nature of the construction in the presence of extraction.
[8] This is not dissimilar to the way a Multi-Component TAG groups together trees into elementary sequences. However, an MCTAG used here would be non-local (see Weir, 1988), and hence formally unconstrained. In addition, the MCTAG would not capture dependency structure.

References

Abney S. (1995) Chunks and Dependencies: Bringing Processing Evidence to Bear on Syntax. In *Computational Linguistics and the Foundations of Linguistic Theory*, CSLI.

Aho A. V., Ullman J. D. (1969) Syntax Directed Translations and the Pushdown Assembler. *J. Comp. Syst. Sci.*, 3/1, pp. 37–56.

Anderson J. (1971) *The Grammar of Case: Towards a Localistic Theory*. Cambridge University Press, Cambridge, UK.

Bauer L. (1979) Some Thoughts on Dependency Grammar. *Linguistics*, 17, pp. 301–315.

Becker T., Joshi A., Rambow, O. (1991) Long Distance Scrambling and Tree Adjoining Grammars. In *Fifth Conference of the European Chapter of the Association for Computational Linguistics (EACL'91)*, pp. 21–26.

Candito M.-H., Kahane S. (1998) Defining dtg Derivations to Get Semantic Graphs. In *Proceedings of the TAG+4 Workshop*, University of Pennsylvania, August, pp. 25–28.

Chiang D., Schuler W., Dras M. (2000) Some Remarks on an Extension of Synchronous TAG. In *Proceedings of TAG+5*, Paris, France, pp. 61–66.

Chomsky N. (1986) *Knowledge of Language*. Praeger, New York, NY.

Collins M. (1997) Three Generative, Lexicalised Models for Statistical Parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL '97)*.

Covington M. (1984) *Syntactic Theory in the High Middle Ages*. Cambridge University Press, London, UK.

Dras M. (1999a) A Meta-Level Grammar: Redefining Synchronous TAG for Translation and Paraphrase. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL '99)*, pp. 80–87.

Dras M. (1999b) *Tree Adjoining Grammar and the Reluctant Paraphrasing of Text*. PhD thesis, Department of Computing, Macquarie University.

Dras M., Bleam T. (2000) How Problematic are Clitics for S-Tag Translation? In *Proceedings of the Fifth International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+5)*, Paris, France, pp. 241–244.

Frank R. (1992) *Syntactic Locality and Tree Adjoining Grammar: Grammatical Acquisition and Processing Perspectives*. PhD thesis, Computer Science Department, University of Pennsylvania.

Frank R. (2000) *Phrase Structure Composition and Syntactic Dependencies*, unpublished manuscript, Johns Hopkins University, October.

Gaifman H. (1965) Dependency Systems and Phrase-Structure Systems. *Information and Control*, 8, pp. 304–337.

Gladkij A. (1980) Opisanie sintaksičeskoj struktury preloženija s pomošč'ju sistem sintaksičeskix grupp. *Slavica*, 17, pp. 5–38.

Gorn S. (1962) Processors for Infinite Codes of Shannon-Fano Type. In *Symp. Math. Theory of Automata*.

Hays D. (1964) Dependency Theory: A Formalism and Some Observations. *Language*, 40, pp. 511–525.

Hudson R. (1976) *Arguments for a Non-Transformational Grammar*. Chicago University Press, Chicago, IL.

Hudson R. (1984) *Word Grammar*. Basil Blackwell, Oxford, UK.

Iordanskaja L., Kim M., Kittredge R., Lavoie B., Polguhre A. (1992) Generation of Extended Bilingual Statistical Reports. In *Proceedings of the 15th International Conference on Computational Linguistics*, Nantes, France, pp. 1019–1023.

Joshi A. (2000) Relationship between Strong and Weak Generative Power of Formal Systems. In *Proceedings of the Fifth International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+5)*, Paris, France, pp. 107–114.

Joshi A., Becker T., Rambow O. (2000) Complexity of Scrambling: A New Twist to the Competence–Performance Distinction. In *Tree Adjoining Grammars: Formalisms, Linguistic Analysis and Processing*, CSLI Publications, Stanford, California.

Joshi A., Vijay-Shanker K. (1999) Compositional Semantics with Lexicalized Tree-Adjoining Grammar (LTAG): How Much Underspecification is Necessary? In *Proceedings of the 2nd International Workshop on Computational Semantics*.

Kahane S., Nasr A., Rambow O. (1998) Pseudo-Projectivity: A Polynomially Parsable Non-Projective Dependency Grammar. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics (ACL '98)*, Montreal, Canada.

Kasper R., Kiefer B., Netter K., Vijay-Shanker K. (1995) Compilation of hpsg to Tag. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL '95)*.

Kayne R. S. (1994) *The Antisymmetry of Syntax*. MIT Press, Cambridge, MA.

Kroch A. S., Joshi A. K. (1987) Analyzing Extraposition in a Tree Adjoining Grammar. In Huck G., Ojeda A. (eds.), *Discontinuous Constituents, Syntax and Semantics*, Vol. 20, Academic Press.

Kulick S. (2000) A Uniform Account of Locality Constraints for Clitic Climbing and Long Scrambling. In *Proceedings of the Penn Linguistics Colloquium*.

Lecerf Y. (1960) Programme des conflits, modèle des conflits. *Bulletin bimestrial de l'ATALA*, 4.

Magerman D. (1995) Statistical Decision-Tree Models for Parsing. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL'95)*.

Mel'čuk I. (1979) *Studies in Dependency Syntax*. Karoma Publishers, Ann Arbor, MI.

Mel'čuk I. (1988) *Dependency Syntax: Theory and Practice*. State University of NY Press, Albany.

Mel'čuk I., Pertsov N. (1987) *Surface Syntax of English: A Formal Model within the Meaning-Text Framework*. John Benjamins, Amsterdam, The Netherlands.

Palmer M., Rosenzweig J., Schuler W. (1998) Capturing Motion Verb Generalizations with Synchronous TAG. In Dizier P.S. (ed.), *Predicative Forms in NLP*, Kluwer Press.

Peters P., Ritchie R. (1973) On the Generative Power of Transformational Grammars. *Information Sciences*, 6, pp. 49–83.

Rambow O. (1994) *Formal and Computational Aspects of Natural Language Syntax*. PhD thesis, Computer Science Department, University of Pennsylvania.

Rambow O., Joshi A. (1997) A Formal Look at Dependency Grammars and Phrase-Structure Grammars, with Special Consideration of Word-Order Phenomena. In Wanner L. (ed.), *Current Issues in Meaning-Text Theory*, Pinter, London.

Rambow O., Satta G. (1996) Synchronous Models of Language. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL '96)*.

Rambow O., Weir D., Vijay-Shanker K. (1995) D-Tree Grammars. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL '95)*.

Robinson J. (1970) Dependency Structures and Transformational Rules. *Language*, 46/2, pp. 259–285.

Rogers J. (1994) Capturing CFLs with Tree Adjoining Grammars. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics (ACL '94)*.

Rogers J. (1997) A Unified Notion of Derived and Derivation Structures in Tag. In *Proceedings of the Fifth Meeting on Mathematics of Language (MOL5)*, Dagstuhl, Germany.

Schabes Y., Shieber S. M. (1994) An Alternative Conception of Tree-Adjoining Derivation. *Computational Linguistics*, 20/1, pp. 91–124.

Schuler W. (1999) Preserving Semantic Dependencies in Synchronous Tree Adjoining Grammar. *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL '99)*.

Sgall P., Hajičová E., Panevová J., Mey J. L. (eds.) (1986) *The Meaning of the Sentence in its Semantic and Pragmatic Aspects*. Academia, Prague.

Shieber S. (1985) Evidence Against the Context-Freeness of Natural Language. *Linguistics and Philosophy*, 8, pp. 333–343.

Shieber S. M. (1994) Restricting the Weak-Generative Capability of Synchronous Tree Adjoining Grammars. *Computational Intelligence*, 10/4.

Srinivas B. (1997) *Complexity of Lexical Descriptions and its Relevance to Partial Parsing*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania.

Tesnière L. (1959) *Éléments de syntaxe structurale*. Librarie Klincksieck, Paris.

Vater H. (1975) Toward a Generative Dependency Theory. *Lingua*, 36, pp. 121–145.

Vijay-Shanker K. (1987) *A Study of Tree Adjoining Grammars*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania.

Vijay-Shanker K., Weir D. (1993) The Use of Shared Forests in Tree Adjoining Grammar Parsing. In *Proceedings of EACL '93*, pp. 384–393.

Weir D. (1988) *Characterizing Mildly Context-Sensitive Grammar Formalisms*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania.

XTAG Research Group (1998) A Lexicalized Tree Adjoining Grammar for English. Technical report, University of Pennsylvania.