

Rambow, Vijay-Shanker, Weir 1995: D-Tree Grammars

They visit the problem that substitution and adjunction don’t map cleanly onto complementation and modification (this goes back to Schabes and Shieber 1994, if not earlier), and the problem of complementation modification dependencies going in the wrong direction.

(1) Small spicy hotdogs he claims Mary seems to adore.

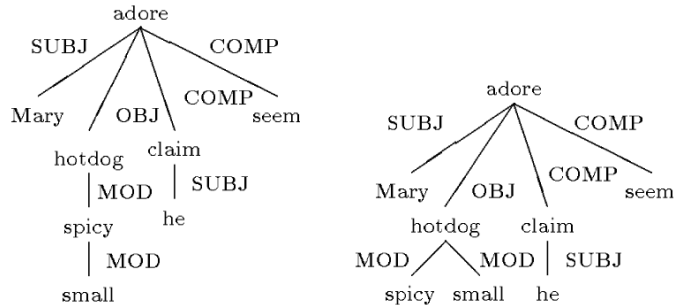


Figure 1: Derivation trees for (1): original definition (left); Schabes & Shieber definition (right)

Frank, Kulick and Vijay-Shanker 2000: Monotonic C-Command: A New Perspective on Tree Adjoining Grammars

Some syntactic relations can’t be established in plain TAG:

(2) Does Gabriel seem to eat gnocchi?

Bob wants ‘does’ and ‘seem’ to be in the same tree to accord with his CETM principle, since ‘seems’ projects up to CP. They can’t be part of the same elementary tree if ‘Gabriel’ and ‘eat’ are. That would require some sort of interleaving.

Besides the CETM, how do we know that there is a syntactic relation between ‘does’ and ‘seem’? Crucially (at least according to me, or whoever raised this in the TAG class a few years ago), ‘seem’ selects for ‘do’ whereas ‘likely’ selects for ‘is’:

(3) Is Gabriel likely to eat gnocchi?

And so there is a syntactic dependency that has to be realized somehow between the auxiliary and the verb, either through a local relation within a tree or a local relation between two trees that are directly attached to one another. Similarly, the auxiliary specifies the tense of the semantically highest verb (‘seem’), not the verb at the root of the TAG derivation tree (‘eat’). However, since a clause can only have one tense, this is not immediately problematic (see below).

Frank also notes:

(4) To whom does Gabriel seem *t* to eat gnocchi?

Even if we don’t need ‘does’ and ‘seem’ to be in the same elementary tree or in a local relation, we very strongly want ‘to whom’ and ‘seem’ to be, and this raises the same issue.

Candito and Kahane 1998: Can the TAG derivation tree represent a semantic graph? An answer in the light of Meaning-Text Theory

The arcs in a TAG derivation tree correspond to semantic dependencies, in a uniform way, but with adjunction being a dependency in the opposite direction compared to substitution. The adjoined tree is the governor. But not all derivation trees show proper semantic dependencies...

Kallmeyer & Romero 2007 dub “the missing link problem”:

Some semantic relations are not evident in the derivation tree: “in some cases this simple semantic procedure is insufficient since the derivation tree does not provide enough information to correctly construction the desired semantic dependencies.” (Cites also Dras et al 2004, Frank and van Genabith 2001, Gardent and Kallmeyer 2003.)

Two adjunctions to the same tree:

(5) John, Paul claims Mary seems to love.

‘seems’ adjoins to ‘love’, ‘claims’ adjoins to ‘love’. There is no connection between claims and seems in the derivation tree. And, further, how is it determined that the semantics is claims >> seems, and not seems >> claims?

Predicate adjunction (versus modifier adjunction):

(6) Who does Paul think John said Bill liked?

‘think’ adjoins to ‘said’ which adjoins to ‘liked’. Here the problem is establishing the correct scope of the wh operator. The wh operator must scope immediately above ‘think’, and so we might want either ‘who’ or ‘liked’ to be in a local relation with ‘think’, but no such relation exists.

So what dependencies do we want?

Note that this makes us (or at least me) rethink what *should* be at the top of a semantic dependency structure. For instance, with raising and bridge verbs, we find that the TAG derivation tree does not agree with our usual notion of semantic dependencies. TAG places the innermost verb, or the one that extraction is out of, at the root of the derivation tree.

TAG	SEMANTICS
love	claims(z, seems(love(y,x)))
seems	WH(claims(z, seems(love(y,x_WH))))
claims	

But no! When we have wh-extraction, we know there is some sort of question operator at maximal scope, and this question operator *originates* from the innermost verb. So in fact, the thing with maximal scope *is* associated with the verb that had an extracted argument (if it’s associated with a verb at all).

Let’s recall that some issues in determining the directions of dependency arcs are: raising and bridge verbs, standard clausal embedding, determiners as heads of NPs, intersective/non-intersective adjectives, auxiliary verbs to main verbs, adjectives to nouns, ...

We can ask two questions:

1. What do we want the TAG derivation tree to look like so we are happy?
2. On what structural representation does block degree and well-nestedness apply?

The answers to both could be:

- Pure Semantics, i.e. the final PrL expression, but some words contribute multiple parts!
= Not a tree!
- Lexicalized Semantics with words representing the *predicates*
= A classical or naive dependency tree (not that I would know)
- Lexicalized Semantics with words representing their highest scope position (wh operators!)
= TAG Derivation Tree in so far as the innermost verb is on top
- TAG Derivation Tree (with the remaining missing link problems)
- Surface Syntax, e.g. raising verbs and auxiliaries have arcs to subjects

Nivre in “Dependency Grammar and Dependency Parsing” recalls criteria (due to someone else) for determining the directions of dependencies between the head/governor (H) and dependent (D) that form a constituent (C): (p4 in <http://vxu.se/msi/~nivre/papers/05133.pdf>)

1. H determines the syntactic category of C and can often replace C
2. H determines the semantic category of C; D gives semantic specification.
3. H is obligatory; D may be optional.
4. H selects D and determines whether D is obligatory or optional.
5. The form of D depends on H (agreement or government).
6. The linear position of D is specified with reference to H.

We (Joan, Lucas, Josh) find that the criteria give mixed or unclear answers in a lot of cases.

The Multi-Component Approach to the Syntactic Problem:

The multi-component approach to Frank’s syntactic problem proposes a tree set { ‘does’ ‘seem’ } in the lexicon. This tree set adjoins into ‘eat’ with one part going to C’ and one part going to I’.

But what happens when there are two or more raising verbs?

(7) Does John seem to be likely to know French?

There are two approaches:

The *set-local* approach:

{‘does’ ‘seem’} attaches pair-wise to a set with a degenerate component {‘ · ’ ‘to be likely’}, yielding {‘does’ ‘seem to be likely’} and then this set attaches to the initial tree. This maintains the derivation tree structure we are accustomed to.

The *tree-local* approach but with *flexible composition*:

{‘to be likely’} attaches at the foot of the second part of {‘does’ ‘seem’}, yielding {‘does’ ‘seem to be likely’}, which then attaches to the initial tree. However, this flirts with a new derivation tree and an undesirable dependency structure:

likely ← seems ← know

Kallmeyer 2002: Using an Enriched TAG Derivation Structure as Basis for Semantics

Kallmeyer 2002: Enriching the TAG Derivation Tree for Semantics

Enrich the derivation tree with additional enrichment links. But what machinery is required? When an auxiliary tree A attaches at the root of tree B, a new enrichment edge is created from A to whatever B attaches to and whatever B is connected to via an upward enrichment edge. This makes an ‘e-derivation structure’.

(8) Every dog barks. barks ‘every’ adjoins to the root of ‘dog’

```

      |      \
      dog     \
      |      // <--- new enrichment edges
      every
  
```

This is also useful for unboundedly embedded interrogatives, where the interrogative is syntactically embedded but semantically takes a much wider scope:

(9) Mary wondered who Peter thought John said bill liked.
 wonder(mary, **who**(x, think(peter, say(john, like(bill, x))))))

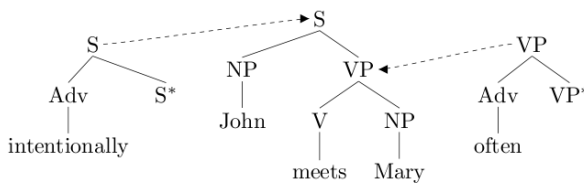
Enrichment links are added that establish a relation between ‘think’ and ‘who’. Note that this is about the *minimum* scope of the question operator, not the maximum scope. It must outscope ‘think’.

This also solves the problem of “roasted red peppers”. Traditionally, ‘roasted’ adjoins to ‘red’ and ‘red’ to ‘peppers’. But if the adjectives are intersective, a direct relation between ‘roasted’ and ‘peppers’ is also desired. Since ‘roasted’ adjoins to the root of ‘red’, an enrichment link between ‘roasted’ and ‘peppers’ is created.

Additional links do not fix the *two adjunctions to the same tree* case, because neither adjunction is at the root. However, Kallemeier proposes that the adjuncts be treated as, effectively, intersective, like adverbs.

Gardent & Kallemeier 2003: Semantic construction in Feature-Based TAG

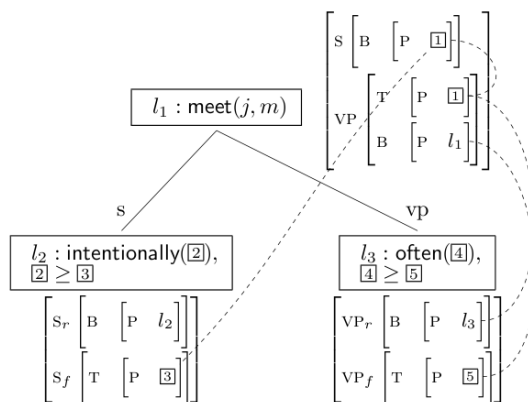
Kallemeier & Romero 2005



Uses (roughly) feature TAG with ‘top’ and ‘bottom’ features. The features are tied to the semantics of each tree with feature variables (boxed numbers).

(10) Intentionally, John often meets Mary.

The VP spine moderates the order of predicates. The spine has a single “top” variable which auxiliary trees can effectively rewrite as they adjoin in.



Selectional Restrictions of Bridge Verbs:

A related problem that Josh raised is that *some*^{ure 11}. Modifiers at S and VP: missing link problem
 bridge verbs (ECM verbs like ‘believe’) allow the next embedded verb to be either finite or non-finite, while others (like ‘think’, ‘say’, and ‘shout’) require the verb in the same position to be finite. Thus there is some sort of dependency, that we would like to be local, between the bridge verb and whatever tree introduces tense into the embedded clause

(11) John believes Mary {is | to be} likely to win the lottery.
 (12) John thinks Mary {is | *to be} likely to win the lottery.

Here we would not postulate a tree set { ‘believes’ ‘likely’ }! However, we have not carefully looked at the TAG derivations in these cases. Also, it is worth checking if recursive clauses pose a problem because in that case a single sentence may have many tenses, and we can only stuff so many things into the feature structures.

The Feature-Passing Approach to the Syntactic Problem:

It is possible to use features to pass the information around necessary for the does...seems cases (Tanja). Since a clause can only have a single auxiliary ('do' or 'be'), it is easy to arrange for features to be passed from the initial tree to trees that adjoin to it that flip a be-versus-do switch for the whole clause. Similarly, a tense feature could be passed around this way. An AUXTYPE feature would be placed on the I' node and also on the C' node, so that adjunction to I' can control what types of things can adjoin at C'.

No doubt this could also solve the bridge verb issue.

Solving the Syntactic Problem with D-Tree Grammars

(Rambow, Vijay-Shanker, Weir 1995: D-Tree Grammars)

Their proposal is D-Tree grammars, which break up elementary trees into parts and compose them using "subsertion" which allows interleaving. The analysis of (1) is then: 'to adore' subserts into 'seems', which all then subserts into 'claims'.

Solving the Syntactic Problem with Monotonic C-Command Relations

(Frank, Kulick and Vijay-Shanker 2000: Monotonic C-Command: A New Perspective on Tree Adjoining Grammars)

Similar to the D-Tree approach, but uses c-command as the relation between tree parts. Supposedly is more linguistically relevant, ruling out super-raising in a way D-Trees did not.

Solving the Syntactic Problem with Segmented Adjoining

(Kulick 2000: Constraining Non-Local Dependencies in Tree Adjoining Grammar)

Seth proposes a third approach to the problem. (As with the other approaches, it also addresses additional problems.)

Solving the Semantic Problem By Reordering the Derivation Tree

It was our suspicion that enriching the derivation tree with additional links as a means to establish the right semantic dependencies was prematurely abandoned, and some of us (at least Josh, probably Lucas) believe that while the feature-based approach to TAG semantics may descriptively capture the facts, it lacks or obscures the explanatory generalization of what is going on. This re-raises the old question: what can we do to the derivation record to establish the semantic dependencies?

The answer appears to be a post-order traversal of the derivation tree, subject to the following notes:

- Only adjunction edges are traversed. Substitution edges are presumed to remain hanging below the nodes they attach to, but are left for future work.
- The derivation tree is ordered left-to-right according to the node addresses at which the adjunction occurs. We are only concerned with adjunction on the spine of a tree, so we say the nodes are ordered top-left to bottom-right. Adjunction off of the spine is left for future work.

Taking sentence (13), combining most of what we've seen so far:

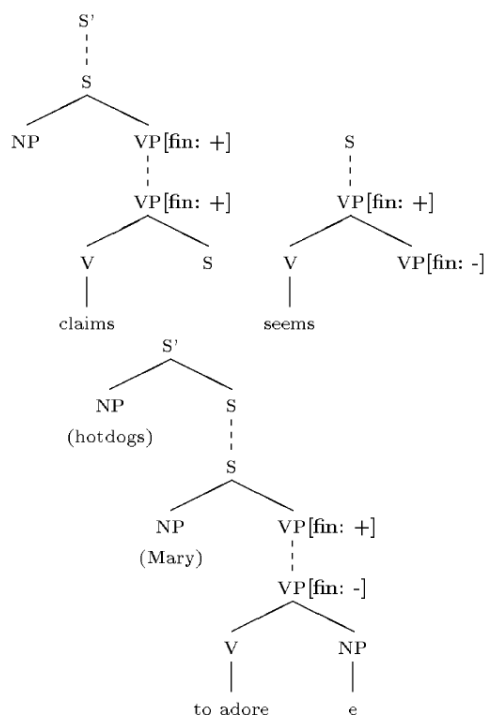


Figure 4: D-trees for (1)

(13) Mary knows what Paul claims roasted red peppers seem to be likely to resemble.

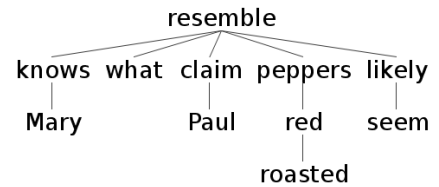
whose derivation tree is to the right, the post-order traversal of adjunction edges yields what we want:

knows, claim, seem, likely, resemble.

Adding the substitution dependencies back in yields:

knows(mary, claim(paul,
seem(likely(roasted(red(peppers)), what).

which is pretty good. Note the post-order traversal in the NP too. Only the question semantics is wrong.



Quantified NPs

Kallmeyer set the precedent for looking at the scope of quantified NPs in terms of the missing link problem with “every dog barks”, using enriched trees. Though this was of course later abandoned in favor of features. But we can give a brief treatment to determining the maximum scope of QuNPs in terms of the post-order traversal semantics. Take:

(14) John believes many people to be lucky. / John believes many people are lucky.

whose post-order traversal is believes >> lucky. To find the maximum scope of the QuNP, start with the tree that the QuNP substitutes into: lucky. Then go up the post-order order and stop after you hit the first +tense verb. This captures the gross generalization the QuNPs can scope only out of untensed clauses.

Note that while adverbs might be a part of the post-order traversal, they definitely don't have +tense.

Question Semantics

Kallmeyer also raised the question of how the scope of question operators gets set. In (13), we want to say that the question operator associated with ‘what’ should minimally scope over ‘claim’, and maximally under ‘knows’.

There is a sense in which we can read this off of the derivation tree: ‘what’ appears in the tree, based on its node position, between ‘knows’ and ‘claim’. Thus, do we really want to exclude NPs from the post-order traversal that gives us the semantic dependencies? Considering that QuNPs make two contributions to the semantics (the outer predicate and the bound variable), it seems clear that something more is going to have to be said one way or another.

Other Notes

Lucas raises the point that perhaps Binding Condition C, for long distance reflexives, is another case of a missing link.