

## Empirical cases that rule out Ternary Merge

Ternary Merge both *undergenerates* and *overgenerates* massively.

First. **Undergeneration.** There are simple cases that cannot be generated by ternary merge. Obvious examples are *It rains, John arrived, John looked at her* (versus *John watched her*), *John has watched her* (versus *John has looked at her*, i.e. {*John, has, {looked, at, her}*}), and *Who has John looked at who* (versus *Who has John watched*, i.e. {*Who, has, {John, watched, who}*}), etc. These cases of undergeneration can be produced indefinitely. Elements with “strike-through” are copies that are deleted in externalization.

Of course, a lot depends on how derivations proceed under current theories (e.g. assumptions about functional categories, agreement, labeling, and phases) but it’s ALWAYS possible to show undergeneration under ternary merge once we agree upon some set of underlying assumptions of internal language derivations.

All these problematic cases of undergeneration can be SIMPLY overcome by assuming that any ternary merge system additionally allows binary merge (i.e. in general, *n*-ary merge entails (*n*-1)-ary Merge. However, since every ternary merge-based structure can be alternatively reformulated as the result of a sequence of binary merge applications, why should we accept ternary Merge in the first place? *It’s not necessary (conceptually, empirically), and it’s not sufficient (empirically) either.*

But there’s more. Ternary Merge fails in another respect, namely **Descriptive Adequacy**. Application of Ternary Merge (assumed to be *legitimate*) yields *illegitimate* results with lethal consequences. These results are the problematic “empirical cases” that Robert requested, I think. They fall under the next rubric.

Second. **Overgeneration.** Consider the ungrammatical sentences (1)-(6), which are merge-generated under ternary merge but violate all kinds of “island conditions” covered by binary merge-derivation.

- (1) *peanuts monkeys children will throw*  
cf. *children will throw monkeys peanuts*
  - (2) *who do you wonder what they believe likes*  
cf. *You wonder what they believe Mary likes*
  - (3) *who have you been reading books which we think impressed*  
cf. *You have been reading books which we think impressed John*
  - (4) *which cruel pictures does she think that showing to children upsets him*  
cf. *She thinks that showing cruel pictures to children upsets him*
  - (5) *what do we know who she thinks complying with annoys*  
cf. *We know who she thinks that complying with rules annoys*
  - (6) *who does Mary realize what John will tell that we ruined*  
cf. *Mary realizes what John will tell Ann that we ruined*
- Etc..

To show that these ungrammatical sentences are ruled “grammatical” under ternary merge law, we must first see how simple, i.e. binary, merge operates.

Assume the workspace  $WS = [ a, b, \{c, d\} ]$ . Binary Merge selects two terms from  $WS$ , e.g. “*b*” and “ $\{c, d\}$ ,” and constructs a new set containing just these terms, i.e.  $\{b, \{c, d\}\}$ , while removing “*b*” and “ $\{c, d\}$ ” from the newly constructed workspace  $WS'$ . Removal follows from minimal computation requirements that are necessary for organic systems (i.e. derivations must terminate on empty  $WS$ ). The terms removed (i.e. “*b*” and “ $\{c, d\}$ ”) persist in the newly merge-generated construct  $\{b, \{c, d\}\}$ .

**Merge(*b*,  $\{c, d\}$ ,  $WS$ ) =  $WS' = [ \{b, \{c, d\}\}, a ]$ .**

Workspace  $WS = [ a, b, \{c, d\} ]$  has 5 accessible terms (namely,  $a, b, c, d$ , and  $\{c, d\}$ ). The newly constructed  $WS'$  has 6 accessible terms (namely,  $a, b, c, d, \{c, d\}$  and  $\{b, \{c, d\}\}$ ). Merge must do something but must not do too much. It minimally increases the number of terms available for further merge computation by exactly 1 term. This is a strong resource restriction **RR** on Merge applying to  $WS$ , guaranteeing minimal computation.

Merge( $X, Y, WS$ ) has two cases.  $X$  and  $Y$  may be disjoint terms. This case, illustrated above, is External Merge (**EM**), and generates contiguous structures. Or one term is a term of the other. This case is Internal Merge (**IM**) and generates “displacement” effects. This is illustrated below. Let’s assume  $WS$  is  $[ a, \{b, c\}, d ]$ . Again, **RR** is satisfied: accessible terms are increased by exactly 1 since the “lower” copy “ $c$ ” is protected from accessibility by the higher copy. One one copy counts.

**Merge( $c, \{b, c\}, WS$ ) =  $WS' = [ \{c, \{b, c\}\}, a, d ]$ .**

This is all there’s to know for our purposes. Merge is a computation on  $WS$  and is restricted by **RR**.

**ASIDE** This restriction on resources **RR** is crucial, ensuring that terms of computation be disjoint and excluding “overlapping” constituents on principled grounds. Merge and **RR** yield specific grammars that disallow e.g. graph-theoretic approaches (technically, Parallel Merge is excluded). E.g. it cannot be the case that “with” in *“Comply with this”* can simultaneously be a part of  $V$  (i.e. “comply with” as in *frequently complied-with rules*) and a part of  $PP$  (i.e. “with this” as in *rules with which to comply*). This is a major result.

Suppose we try to derive just this. Assume workspace  $WS = [ \{with, this\}, comply, \dots ]$ . Let Merge select from  $WS$  two terms, e.g.  $P = \text{“comply”}$  and  $Q = \text{“this,”}$  and construct a set,  $\{P, Q\}$  from these. This is what Merge does. We thus have **Merge( $P, Q, WS$ ) =  $[ \{comply, with\}, \{with, this\}, \dots ] = WS'$** . The number of terms accessible for further computation has been increased by #2 in  $WS'$  relative to  $WS$ . The Merge result is illegitimate and excluded by **RR**.

If  $WS$  had #4+n accessible terms (namely *with, this, {with, this}, comply*, and  $n$  terms in ....), then  $WS'$  has #6+n accessible terms (namely *comply, with, {comply, with}, with, this, {with, this}*, and  $n$  terms in ....). Since narrow Merge is constrained by **RR** to increase accessibility by #1, this condition minimizes the computational load that Merge imposes on  $WS$ . The condition suffices to rule out illegitimate results of legitimate derivations of broader merge not **RR**-constrained.

Now onto **Ternary Merge**. Assume workspace  $WS = [ d, e, \{a, b, c\} ]$  with three members, i.e. the syntactic objects  $SO$  “ $d$ ”, “ $e$ ” and “ $\{a, b, c\}$ ,” the latter  $SO$  a ternary set constructed by a previous merge operation. [  $WS$  has three members but 6 terms that are accessible for computation, namely  $d, e, a, b, c$ , and  $\{a, b, c\}$ . ]

We illustrate three cases of ternary Merge applying to  $WS$ . Merge selects three terms and constructs a new set containing them, removing the selected terms from  $WS'$ , analogously to binary merge.

- Merge( $a, b, \{a, b, c\}, WS$ ) =  $[ \{a, b, \{a, b, c\}\}, d, e ]$  Internal Merge (IM)
- Merge( $d, e, \{a, b, c\}, WS$ ) =  $[ \{d, e, \{a, b, c\}\} ]$  External Merge (EM)
- Merge( $a, d, \{a, b, c\}, WS$ ) =  $[ \{a, d, \{a, b, c\}\}, e ]$  Hybrid Merge (HM)

Finally, we’ll illustrate how ungrammatical (“illegitimate”) cases such as (1)-(6) above are generated by means (Ternary Merge) that are hypothesized to be “legitimate.” Ternary Merge generates impossible languages. Not a good operation. In contrast to cases of undergeneration, these cases of overgeneration cannot be avoided by allowing additional operations of binary Merge. Naturally, we

could perhaps try to impose additional constraints excluding or filtering derivations that result in (1')-(6'). But the point is, all this additional and ad hoc machinery is doing is yielding results that come free under binary merge. No hope for ternary merge, it seems.

Ternary Merge is neither sufficient, nor necessary, and fails in descriptive adequacy.

For reasons of space, we'll only give the derivational results rather than the derivations themselves. Ternary IM/EM/HM yields the following derivations for (1)-(6). Strike-through terms are merge-generated copies that are not expressed in externalization. In general, only the highest copy is spelled out in externalized language.

- (1') {peanuts, monkeys, {children, will, {throw, ~~monkeys~~, peanuts }}}}
- (2') {who, do, {you, wonder, {who, what, {they, believe, {who, what, {who, likes, what }}}}}}
- (3') {who, have, {you, been, {reading, books, {who, which, {we, think, {which, impressed, who }}}}}}
- (4') {Wh-XP, does, {she, think, {Wh-XP, that, {{showing, children, Wh-XP}, upsets, them }}}}
- Here Wh-XP stands for {which, cruel, pictures}
- (5') {what, do, {we, know, {what, who, {she, thinks, {what, who, {{complying, with, what}, annoys, who }}}}}}
- (6') {who, does, {Mary, realize, {who, what, {John, will, {tell, who, {what, that, {we, ruined, what }}}}}}}}

These are instances of homogeneous ternary merge-based derivations, in full accord with current ideas about labeling and phase theory that are neutral between binary and ternary Merge.

Some of these cases argue against **Dependency Grammar DG**. We'll not elaborate. Instead, let me conclude with a puzzle for adherents of DG. Assume we have a language that licenses topicalization. That is, expressions like, e.g., *That result we are always looking for!* Now, assume further that this language allows topics to go unexpressed, i.e. the language allows *We are always looking looking for!* to be grammatical with Zero Topic ZT. Finally, assume this language to be Verb-Second. I.e. the finite tensed verb of root clauses always sits in second structural position so that *Yesterday left John for London* would be grammatical. Does such a language exist? Yes, German and Dutch. In Dutch, the topicalized sentence above will be expressed with ZT as *Kijken we altijd naar!* (lit. "Look we always at"). A perfectly normal sentence of a type frequently used.

*[ZT] kijken we altijd naar copy<sub>ZT</sub>* (Cf. *Daar kijken we altijd naar!*)

How would DG analyze this case, explaining its semantics, its linear order, and syntax without the use of hierarchical structure or "empty categories"? I would suggest that DG fails on principled grounds. But you can prove me wrong.

## Further Considerations.

**Binarity** is a pervasive property of language playing a defining role in all of its components. It's a defining property of principles and rules.

**Control.** The subject of “try” controls (i.e. is also functioning as) subject of “go” in *John tries to go* with structure [ *John tries [ X to go ]* ], and the object of “persuade” controls subject of “leave” in *John persuaded Bill to leave* with structure [ *John persuaded Bill [ X to leave ]* ]. The subject of “leave” cannot be both “John” and “Bill”. Compare ungrammatical \* [ *John persuaded Bill [ X to meet ]* ], where “meet” requires a plural subject. However, “John” and “Bill” can't be controlling the subject X. I.e. the empirical absence of split control is excluded, and explained under binary **Control(PRO, X)** but split control is expected under ternary **Control(PRO, X, Y)**.

**Binding.** Reflexives (e.g. “herself”) must find a local binder but cannot have split antecedents. We get *Jane protected Bill from herself* or *Jane compared Bill to herself* but not \**Jane protected Bill from themselves/each other* or \**Jane compared Bill to themselves/each other*. The latter cases would all be incorrectly expected to be grammatical under ternary **Bind(REFL, X, Y)**. However, they are correctly ruled out under binary **Bind(REFL, X)**.

Furthermore, under ternary structure we would also expect both of \**Each other's pictures upset them* and *They upset each other's friends* to be fine since there's no principled way to distinguish between {*e.o's pictures, upset, them*} and {*they, upset, e.o's friends*}. Both *they* and *them* have precisely the same structural properties relative to reciprocal *each other*. In contrast, binary structure does give the proper distinction, namely \*{{*each other's, pictures*}, {*upset, them*}} vs. {*they, {upset, {each other's, friends}}*}. Only in the latter structure does “each other” find a binder “higher” up in hierarchical structure. More precisely, subject *they* c-commands anaphoric *each other*, object *them* does not. Here, term X asymmetrically c-commands term Y iff. Y is contained in Z, Z a sister of X in {X, Z}. Anaphoric elements (reflexives, reciprocals) must be asymmetrically bound by a local antecedent. Binding is handled correctly under binary structure but yields empirically incorrect results under ternary structure.

**Theta Structure.** Each argument gets assigned a single semantic function from its predicate, and each semantic function gets assigned to a single argument. E.g. theta assignments **Rely-on(AGENT, John)** and **Rely-on(THEME, Bill)** yield *John relies on Bill*. But we do not have \**John relies on* meaning “John relies on himself,” something made possible under ternary theta assignment, e.g. **Rely-on(John, AGENT, THEME)**. Actually, theta role assignment presupposes notions like “subject” and “object.” Here AGENT goes with “subject,” and THEME with “object.” But the notions of subject and object, which you get for free under binary branching, i.e., Subject = EA = External (to VP) Argument vs. Object = IA = Internal (to VP) Argument, must be stipulated under ternary branching...

**Agreement.** Similar cases can be constructed for AGREEMENT. Why don't we have \**We invites John?* Ternary structure could not make the distinction unless, again, we stipulate “subject” and “object” and further stipulate that agreement is between subject and verb. Under binary branching, subjects and objects get different treatments that depend on hierarchical position as discussed immediately above. Same asymmetries apply to **Case Assignment** (not illustrated here), i.e. subjects are nominative case-marked, objects are accusative case-marked. Ternary structure cannot determine which is which unless these grammatical functions are stipulated.

**Government.** Cautionary note: the notion of government is eliminated in minimalist grammar but here we use it as a descriptive term for the empirical facts covered by the concept and explained under present formulations of minimalist grammar. There's a special relation of GOVERNMENT holding between a predicate (V, N) and its object, e.g., the relation between “painted”/“painting” and “Aristotle” in *Rembrandt painted Aristotle* or *Rembrandt's painting of Aristotle*. Similarly, we get *Aristotle was painted by Rembrandt* and *Aristotle's painting by Rembrandt*. However, the asymmetry *We believe Mary a genius, Mary was believed a genius* vs. \**Our belief of Mary a genius, \*May's*

*belief a genius* is “unexpected” though obviously empirically correct. Similarly, *John appeared to be rich* vs. \**John appearance to be rich*. Ternary structure could not distinguish between {*painting, Aristotle, by Rembrandt*} and {*belief, Mary, a genius*}. Why is raising of “Aristotle” allowed in the former, and raising of “Mary” blocked in the latter? Ternary branching cannot make the distinction.

With binary branching, we get {{*painting, Aristotle*}, by-Rembrandt } vs. {*belief, {Mary, a-genius}*}. Note that binary structure allows us to distinguish between government holding between “painting” and “Aristotle” (the latter an argument of “painting”) and absence of government between “belief” and “Mary” (the latter an argument of “genius,” not of “belief.” Only governed elements raise, something unexplainable under ternary branching.

**Externalization.** Natural language is primarily hierarchical (universal property) but is linearized in speech (externalization with language-specific parametrization). Externalization yields variety and diversity of language. Just focusing on English, one way of getting externalization done is converting asymmetric c-command (merge-based hierarchical structure) to linear precedence (concatenation-based linearly ordered strings). This is essentially Kayne’s *Linear Correspondence Axiom* (LCA). E.g. in {*SUBJ, {V, OBJECT}*}, SUBJECT asymmetrically c-commands V and OBJECT, and V asymmetrically c-commands OBJECT.<sup>NOTE</sup> Therefore, by transitivity of linear precedence, we get SUBJECT precedes V, and V precedes OBJECT, so that SUBJECT > V > OBJECT.

*LCA:* If X asymmetrically c-commands Y, then X has linear precedence over Y  
E.g., [<sub>VP</sub> NP1 [<sub>VP</sub> V NP2 ] => NP1 > V > NP2

NOTE: For Kayne OBJ is an NP that may branch, e.g. [<sub>NP</sub> [<sub>N</sub> books ] [<sub>PP</sub> about Nietzsche ]], or may not branch, e.g. [<sub>NP</sub> [<sub>N</sub> books]]. In his terms, V asymmetrically c-commands N but not conversely because of the intervening NP that contains N but not V. Noam does away with all of this and gets the job done by root-based morphology. We’ll spare you the finesses.

Ternary structure cannot unambiguously determine linear order. Given {*SUBJ, V, OBJ*}, even if V asymmetrically c-commands SUBJ and OBJ, neither of the latter asymmetrically c-commands the other, and linear precedence cannot be unambiguously determined.

**SOV (Turkish) vs. VOS (Malagasy).** Linear or hierarchical structure? Binary or ternary structure? First, Binding cannot be a universal effect of word order. REFL must *follow* antecedent in SOV, but must *precede* antecedent in VOS. However, these cases are unified if we assume *binary* hierarchical structure. In each case subject S is hierarchically superior to object O. Technically, S asymmetrically c-commands O since the constituent that is a sister to S contains O. A ternary structure could not explain these asymmetries.

Stipulating that only subjects can be antecedents does not explain anything (why subjects?). In fact, notions like subject and object are redundant. Universally, Subject (“External Argument”) is just the name for the higher argument, Object (“Internal Argument”) just the name for the lower argument. Conclusion: Structure (here binary structure) decides the binding issue.

VOS	✓ V REFL Subj	vs.	* V Obj REFL	[[ <sub>VP</sub> V O ] S ]
SOV	✓ Subj REFL V	vs.	* REFL Obj V	[ S [ <sub>VP</sub> O V ] ]

This case illustrates primacy of structure over linear order in narrow syntax (internal language). For further discussion, see the Appendix below.

## APPENDIX on Structure, not strings

Structural principles are universal, linearization/externalization is parametric.

First, (bio)linguists have to explain *evolvability*, *learnability* and *universality* of internal language, and second, they have to account for the *mutability*, *variety*, and *complexity* of externalized language. A new approach to Principles (structure) and Parameters (externalization).

Hierarchical structure is a prevalent and primary property of human language. No language can do without it. We get hierarchical structure for free. It's a result of recursively applied (binary) merge. Linear structure and word order are ancillary, a property of externalized language. Internal language is essentially universal (Basic Property). However, there are various ways to link SM systems to Internal Language for externalization.

Structure and the prevalence of structural over linear adjacency can be found even in morphology. Here we give some examples. I think that some of them pose problems for Dependency Grammar.

### Morphological Case I.

1. \*Polar bears **uninhabited** Antarctica
2. Antarctica is **uninhabited**
3. {un-, {-ed, {inhabit }}}}
4. (2)/(3) => [ un- **[ [inhabit ] -ed ] ]** vs. (1) => [ [ un- **[inhabit ] ]** -ed ]

Comment: “un” and “in” are **linearly** adjacent in both (1) and (2) but are **structurally** adjacent only in (1). Apparently, these affixes cannot be structurally adjacent. In (2), the affix “-ed” structurally intervenes between “un-“ and “inhabit” as shown in (3). The different structures of “uninhabited” in (1) vs. (2) are illustrated in (4). Language is primarily about structure, peripherally about linear order. Structure explains, linear order does not.

### Morphological Case II.

5. “unlockable” ambiguous between (6) and (7)
6. [ [ un- lock ] -able ], i.e. “can be unlocked”
7. [ un [ lock -able ] ], i.e. “cannot be locked”
8. [ un **[ [un- lock ] -able ] ]** vs. \* [ [ un **[ un- lock ] ]** able ]

Comment: Clearly, binary structure is imperative here. Ternary structure explains nothing, binary structure is necessary to account for compositional meaning and explain the semantic ambiguity.

Furthermore, we could go further and make up the adjective “unlockable.” We now explain why “unlockable” could ONLY mean “cannot be unlocked,” i.e. [ un- **[ [un- lock ] -able ] ]**, and can NOT mean “can be locked,” i.e. [ [ un **[ un- lock ] ]** able ].

I guess, dependency grammar could simulate this to some extent. Transitivity of “is dependent on” guarantees some hierarchy. So one might say that “lock” depends on “un-”, which itself depends on “-able” in “unlockable” meaning “can be unlocked,” but the other way around, i.e. “lock” depends on “-able”, which itself depends on “un-” in “unlockable” meaning “cannot be locked.” In brief, one might say: No arc is allowed between “un” and “in” (2) or between two occurrences of “un” in (8). So even the ambiguity of “unlockable” can be “copied” in dependency grammar. But why simulate a explanatory merge-based grammar by a model that cannot explain language?

Dependency grammar faces serious problems in syntax. Here are some empirical cases showing the necessity of (binary) structure for natural language. Things are completely different in syntax where we have to deal with displacement issues. Technically, morphology uses external merge (Merge of

disjoint X, Y), while syntax uses both external and internal merge (Merge of X,Y with one a term of the other).

### Syntactic Case I.

9. **John** wants to support **himself**
10. Ann wants **John** to support **himself**
11. Who do **you** want to support **yourself** (\*) **linear** proximity
12. **Who** do **you** want to support **himself** (ok) **structural** proximity
13. Who do you want [ **who** to support himself ]

⇒ (9) has same analysis as (6), not (5). Also (5) allows contraction (“wanna”), (8) does not since abstract “who” intervenes between “want” and “to.” Compare (9) = (8) with (6).

### Syntactic Case II.

Celtic languages are VSO on the surface and pose an immediate problem. They don't seem to have a VP constituent because of intervening Subject. So no binary structure, apparently. A weird situation for any natural language grammar (Merge-based grammar), though business as usual for dependency grammar. In fact, on deeper inspection, it turns out that VSO languages are well behaved. The deeper structure is actually [V-T [ S [  $\forall$  O ] ] ] with “displaced” V (V searches for Tense and raises to T). Celtic is just a normal family of languages with VP. In fact, we predict that the presence of a tensed auxiliary will block displacement of V. Indeed, the result is that V stays inside VP. So we get a structure like [ AUX [ S [ V O ] ] ]. Semitic languages receive similar analyses. In all these cases, S is hierarchically higher than O, explaining why only S can bind a reflexive O (“John supports himself”) both in English (SVO) and Celtic/Semitic (VSO).

There are more dramatic cases (e.g. VOS). So far, n-ary structure does not explain any of this. Neither does dependency grammar.

There's striking empirical evidence for this. Consider VP-deletion as in “John will buy a house, but Ann already did” (viz. “buy a house”). Question: does VSO have VP deletion? And if it does, how does it look like?

We saw above that Celtic SVO must be analyzed as [TP V-T [VP S [ V O ] ] ] with V raising to T. In contrast, English requires raising of S but not raising of V. The latter rule is additionally required in languages like German, yielding an explanation for why *Left John the house* is an ill-formed question in English (cf. *Did John leave the house*) but is well-formed in German.

- Engl [ **John** [ **can** [VP **John** [ buy a house ] ] ] ] [ **John** [ **doe-s** [VP **John** [ buy a house ] ] ] ]
- Irish [ **can** [VP **John** [ buy a house ] ] ] [ **buy-s** [VP **John** [ buy a house ] ] ]

English VP deletion results in “John can” and “John does.” The subject has been evacuated from VP in English and VP deletion deletes Verb, Object, and copy of Subject. In Celtic the Subject stays put in VP but the V itself has left VP, having been raised to Tense. Therefore, we can make a precise empirical prediction. In Irish, VP deletion deletes VP containing Subject and Object and copy of Verb. I.e. Instead of e.g. “Yes, John does” we expect “Yes, buys.”

The prediction (as expected) turns out to be correct. Compare Irish cases below.

**Chennaigh** siad teach (“They bought a house”)      **Ar cheannaigh** siad teach? (“Did they buy a house?”)  
**Chennaigh** (“Yes, they did”)      **Níor cheannaigh** (“No, they didn't”)