Late Merge as Lowering Movement in Minimalist Grammars

Thomas Graf
tgraf@ucla.edu
tgraf.bol.ucla.edu

Stony Brook University

LACL 2014
July 18, 2012

Topic of This Talk

Concrete Issue: What does Late Merge do?

- Late Merge is too powerful an operation.
- But **as used by linguists**, it can be emulated by a simpler mechanism: **Lowering**.

Bigger Picture: What does Lowering do?

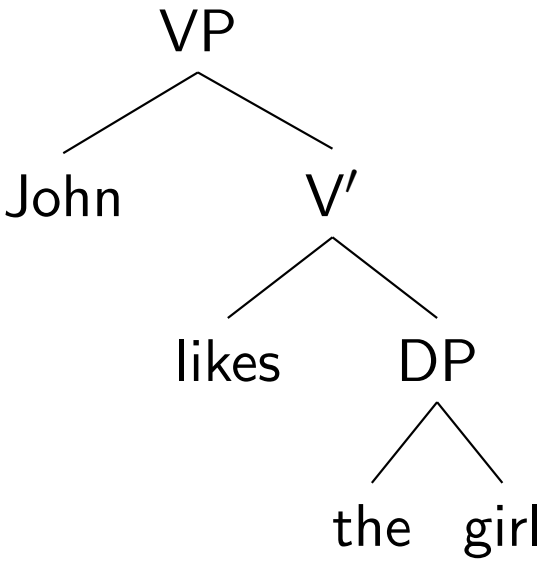
- Lowering by itself is very weak/redundant.
- But: many operations that increase the power of MGs can be captured once Lowering is added.
- How come Lowering it both weak and powerful?

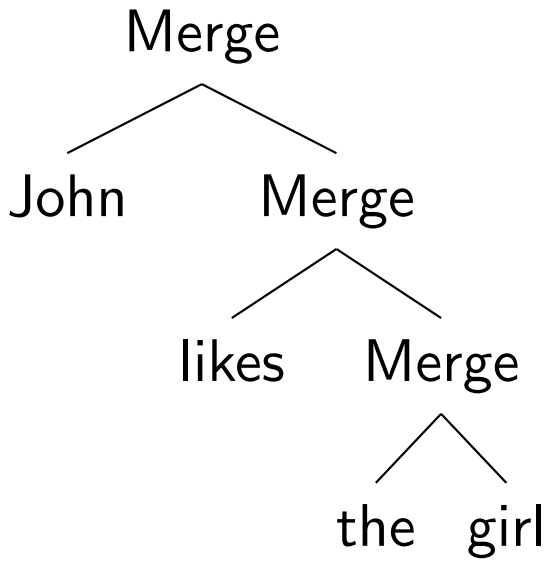
Outline

Minimalist Grammars (MGs)

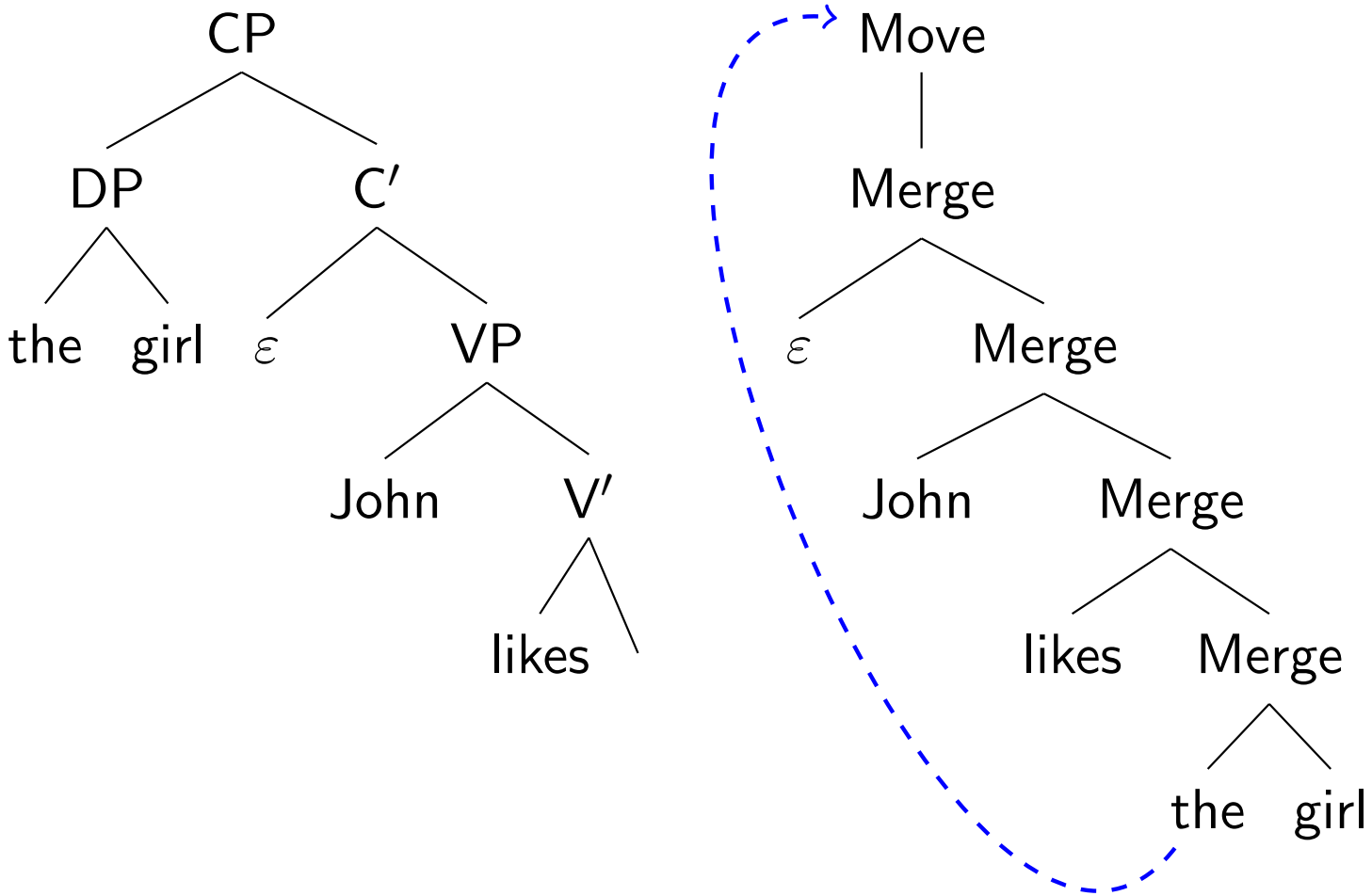
- mildly context-sensitive formalization of Minimalist syntax (Chomsky 1995; Stabler 1997)
- grammar is fully specified by lexicon
- lexicon = finite set of feature-annotated words
- features trigger structure-building operations Merge and Move
- **Merge:** combine two trees into a new tree
- **Move:** move a subtree of tree t to the left of the root of t

Sketch of a Simple Merge Derivation





Sketch of a Derivation with Move



An Important Restriction on Move

Shortest Move Constraint (SMC)

There is some $k \geq 0$ such that at every point of the derivation at most k subtrees have an unchecked movement feature.

- The SMC puts a finite upper bound on how many subtrees can be moved out of a given tree.
- This limit is essential for a variety of formal properties.

Formal Properties of MGs

Automata-Theoretic Results

- MGs $\equiv$ MCFGs (Harkema 2001; Michaelis 2001)
- Every MG's set of well-formed derivation trees is regular. (Michaelis 2001; Kobele et al. 2007)
- The mapping from derivation trees to output trees can be computed by a linear multi bottom-up tree transducer. (Kobele et al. 2007)

Logic Perspective of MGs

MGs are specified via **two model-theoretic components**

- a sentence of monadic second-order logic (MSO) that defines the well-formed derivation trees over a given lexicon,
- an MSO-definable transduction from derivation trees to output trees.

(Morawietz 2003; Mönnich 2006, 2007; Graf 2012a, 2013)

Formal Properties of MGs

Automata-Theoretic Results

- MGs $\equiv$ MCFGs (Harkema 2001; Michaelis 2001)
- Every MG's set of well-formed derivation trees is regular. (Michaelis 2001; Kobele et al. 2007)
- The mapping from derivation trees to output trees can be computed by a linear multi bottom-up tree transducer. (Kobele et al. 2007)

Logic Perspective of MGs

MGs are specified via **two model-theoretic components**

- a sentence of monadic second-order logic (MSO) that defines the well-formed derivation trees over a given lexicon,
- an MSO-definable transduction from derivation trees to output trees.

(Morawietz 2003; Mönnich 2006, 2007; Graf 2012a, 2013)

Movement-Generalized Minimalist Grammars (MGMGs)

MGMGs make it possible to add new movement types to MGs while preserving their core properties. (Graf 2012b)

Basic Idea

MGs do not use all the power of MSO. New movement types may be added as long as **MSO-definability is maintained** for

- the class of well-formed derivation trees, and
- the mapping from derivation trees to output trees.

Two Useful New Movement Types

- Rightward movement
- Lowering (movement to a c-commanded position)

Movement-Generalized Minimalist Grammars (MGMGs)

MGMGs make it possible to add new movement types to MGs while preserving their core properties. (Graf 2012b)

Basic Idea

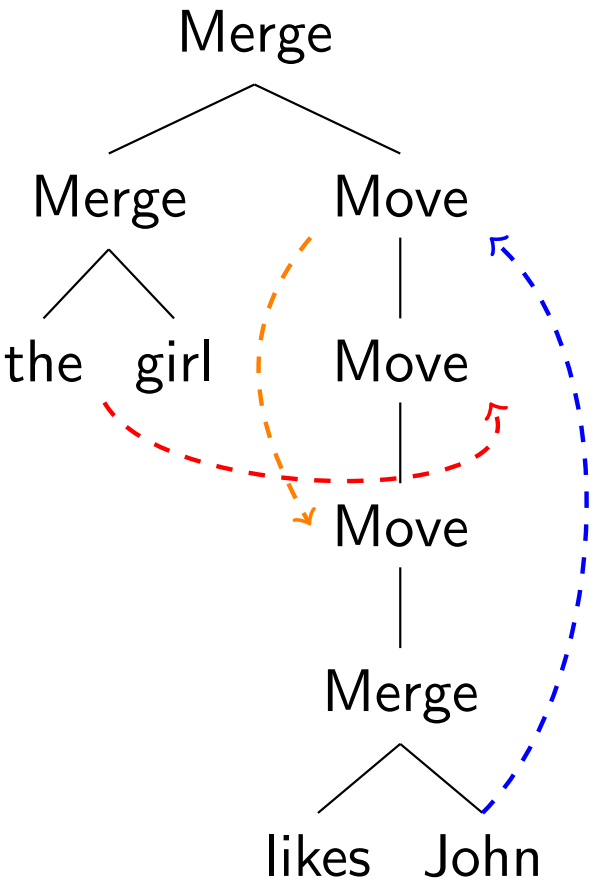
MGs do not use all the power of MSO. New movement types may be added as long as **MSO-definability is maintained** for

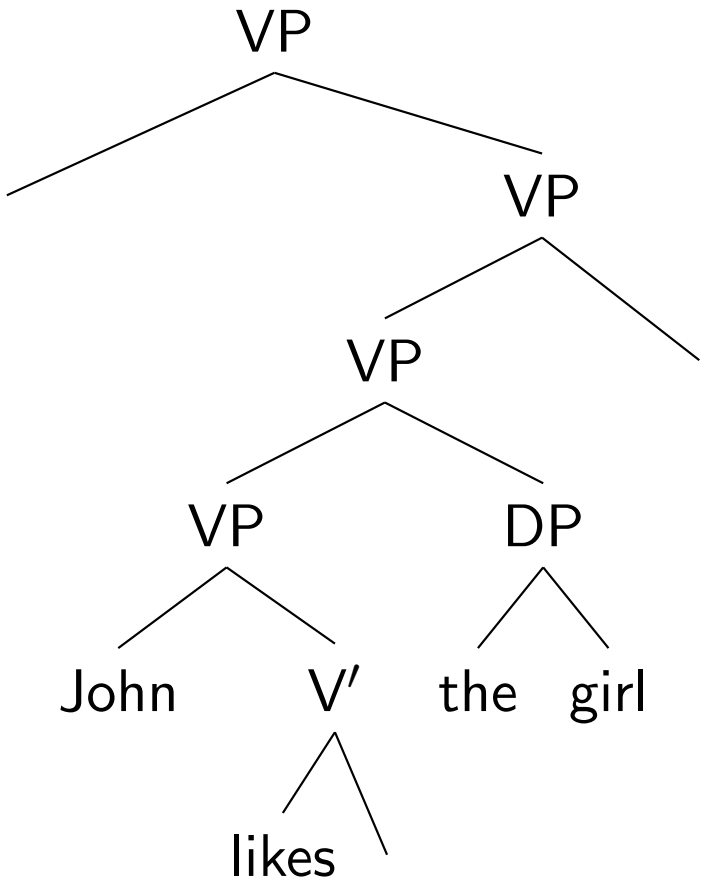
- the class of well-formed derivation trees, and
- the mapping from derivation trees to output trees.

Two Useful New Movement Types

- Rightward movement
- Lowering (movement to a c-commanded position)

Example: Derivation with New Movement Types





Formal Properties of MGMGs

Theorem (Weak Equivalence)

MGs and MGMGs are weakly equivalent.

Proof.

1) MG $\leq$ MGMG $\leq$ str(MSOTT(MSO))
2) MG $\equiv$ MCFG $\equiv$ str(MSOTT(MSO))
$\Rightarrow$ MG $\equiv$ MGMG □

Theorem

MGMGs have greater strong generative capacity than MGs.

Proof.

Every TAG tree language can be generated by some MGMG (Graf 2012c). But the classes of tree languages generated by TAGs and MGs, respectively, are incomparable (Kobele et al. 2007). □

Formal Properties of MGMGs

Theorem (Weak Equivalence)

MGs and MGMGs are weakly equivalent.

Proof.

1) MG $\leq$ MGMG $\leq$ str(MSOTT(MSO))
2) MG $\equiv$ MCFG $\equiv$ str(MSOTT(MSO))
$\Rightarrow$ MG $\equiv$ MGMG □

Theorem

MGMGs have greater strong generative capacity than MGs.

Proof.

Every TAG tree language can be generated by some MGMG (Graf 2012c). But the classes of tree languages generated by TAGs and MGs, respectively, are incomparable (Kobele et al. 2007). □

Interim Summary

- MGs use Merge and Move for building structures.
- Merge combines two trees, Move displaces a subtree.
- SMC: finite bound on number of parallel movers
- Derivation trees are a record of the structure building process.
- Derivation trees are easily mapped to derived trees via MSO.
- MGMGs generalize the mapping = add new movement types

What is Late Merge (Good for)?

Late Merge

Merge is delayed to avoid some constraint violation $\Rightarrow$
material appears in same surface position as with standard Merge, but enters the derivation later

Late Merger of subtree s is used to

- save s from incurring a violation of some constraint
- prevent s from blocking some other operation

Principle C Exceptions

Principle C

An R-expression must not be c-commanded by a coreferent DP.

Adjuncts within a moved phrase can be exempt from Principle C.

(1) a. * **He** believed the argument that **John** made.

b. * Which argument that **John** is a jerk did **he** believe?

c. Which argument that **John** made did **he** believe?

Explanation

- (1a) violates Principle C in the output structure.
- (1b) violates Principle C before movement:

  did **he** believe which argument that **John** is a jerk
- But why is (1c) exempt?

Principle C Exceptions

Principle C

An R-expression must not be c-commanded by a coreferent DP.

Adjuncts within a moved phrase can be exempt from Principle C.

(1) a. * **He** believed the argument that **John** made.

b. * Which argument that **John** is a jerk did **he** believe?

c. Which argument that **John** made did **he** believe?

Explanation

- (1a) violates Principle C in the output structure.
- (1b) violates Principle C before movement:

  did **he** believe which argument that **John** is a jerk
- But why is (1c) exempt?

Principle C Exceptions

Principle C

An R-expression must not be c-commanded by a coreferent DP.

Adjuncts within a moved phrase can be exempt from Principle C.

(1) a. * **He** believed the argument that **John** made.

b. * Which argument that **John** is a jerk did **he** believe?

c. Which argument that **John** made did **he** believe?

Explanation

- (1a) violates Principle C in the output structure.
- (1b) violates Principle C before movement:

  did **he** believe which argument that **John** is a jerk
- But why is (1c) exempt?

Principle C Exceptions [cont.]

- (1b) involves an argument, but (1c) an adjunct.
- Adjunct is **late-merged after the movement step**
  $\Rightarrow$ no c-command $\Rightarrow$ no Principle C violation
  (Lebeaux 1988)

Example Derivation

1. did **he** believe [$_{DP}$ which argument]
2. [$_{DP}$ which argument] did **he** believe
3. [$_{DP}$ which argument [$_{CP}$ that **John** made]] did **he** believe

More Binding: Late Merger of Arguments

Arguments within quantified phrases can also escape Principle C.

(2) a. * Which argument that **John** is a jerk seems to **him** to be false?

b. Every argument that **John** is a jerk seems to **him** to be false

Explanation: Restrictors of quantifiers can also be late-merged. (Takahashi and Hulsey 2009)

Example Derivation

1. seems to **him** [$_{DP}$ every] to be false
2. [$_{DP}$ every] seems to **him** to be false
3. [$_{DP}$ every [$_{NP}$ argument that **John** is a jerk]] seems to **him** to be false

More Binding: Late Merger of Arguments

Arguments within quantified phrases can also escape Principle C.

(2) a. * Which argument that **John** is a jerk seems to **him** to be false?

b. Every argument that **John** is a jerk seems to **him** to be false

Explanation: Restrictors of quantifiers can also be late-merged. (Takahashi and Hulsey 2009)

Example Derivation

1. seems to **him** [$_{DP}$ every] to be false
2. [$_{DP}$ every] seems to **him** to be false
3. [$_{DP}$ every [$_{NP}$ argument that **John** is a jerk]] seems to **him** to be false

English *do*-Support

In English, *do*-support is triggered if the tense marker and the verb are not string-adjacent at some point. (Ochi 1999)

(3) a. John -ed leave ⇒ John left

b. John -ed not leave ⇒ John did not leave

It follows that **VP-adjuncts must be late-merged**, otherwise they would intervene and trigger *do*-support.

Example Derivation

1. John -ed leave
2. John left
3. John quickly left

Subjacency Violations

Subjacency

If position p must be targeted by movement, then
the closest licit mover c-commanded by p must move to p.

Subjacency incorrectly predicts that experiencers of psych verbs should be realized as subjects.

(4) a. It seems to Mary that John is smart.
b. John seems to Mary to be smart.
c. * To Mary seems John to be smart.

Once again this can be fixed by Late Merge. (Stepanov 2001)

Example Derivation

1. seems John to be smart
2. John seems to be smart
3. John seems to Mary to be smart

Subjacency Violations

Subjacency

If position p must be targeted by movement, then
the closest licit mover c-commanded by p must move to p.

Subjacency incorrectly predicts that experiencers of psych verbs should be realized as subjects.

(4) a. It seems to Mary that John is smart.
b. John seems to Mary to be smart.
c. * To Mary seems John to be smart.

Once again this can be fixed by Late Merge. (Stepanov 2001)

Example Derivation

1. seems John to be smart
2. John seems to be smart
3. John seems to Mary to be smart

Subjacency Violations

Subjacency

If position p must be targeted by movement, then
the closest licit mover c-commanded by p must move to p.

Subjacency incorrectly predicts that experiencers of psych verbs should be realized as subjects.

(4) a. It seems to Mary that John is smart.
b. John seems to Mary to be smart.
c. * To Mary seems John to be smart.

Once again this can be fixed by Late Merge. (Stepanov 2001)

Example Derivation

1. seems John to be smart
2. John seems to be smart
3. John seems to Mary to be smart

Formal Properties of MGs with Late Merge

Adding Late Merge to MGs increases

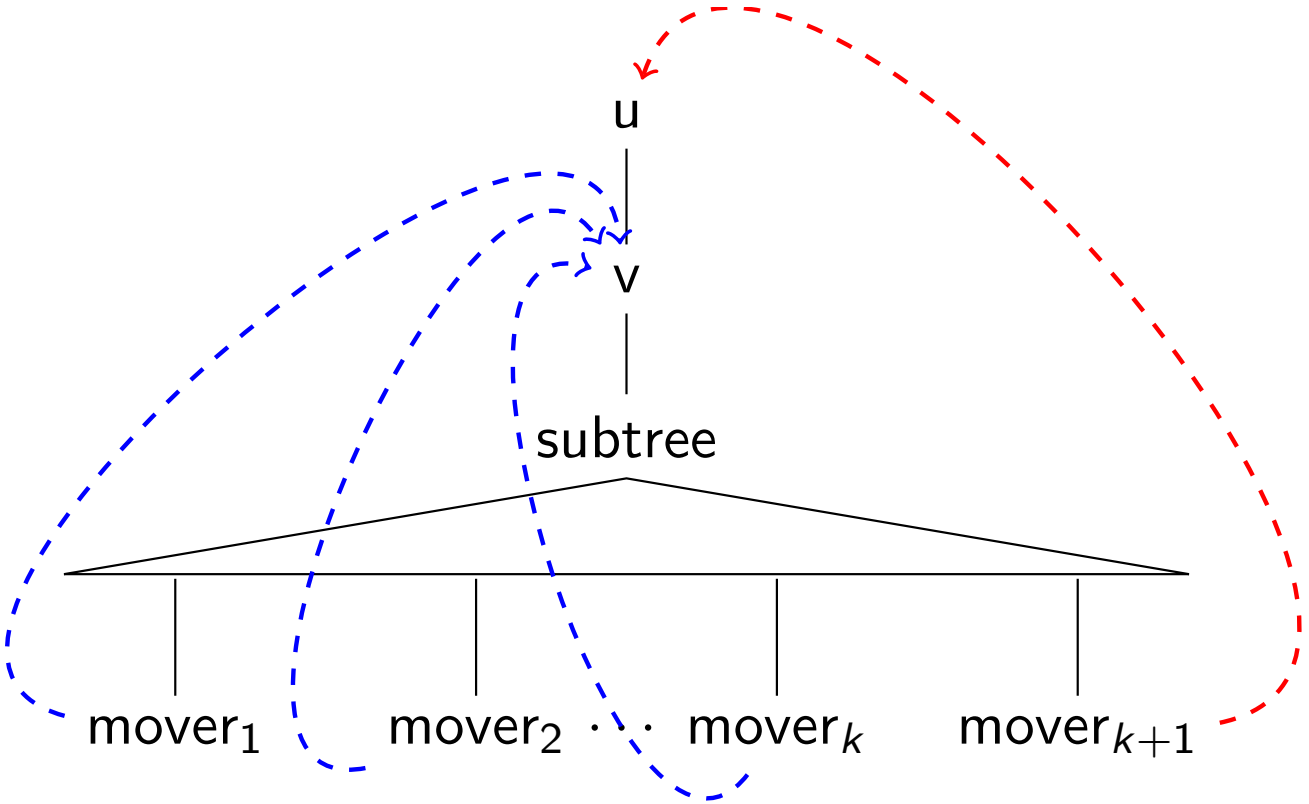
- weak generative capacity (Kobele and Michaelis 2011)
- strong generative capacity (corollary)
- complexity of mapping from derivations to derived trees (Gärtner and Michaelis 2008; Kobele 2010)

Intuition

- Just like Late Merge can escape Principle C violations, it also creates loop holes for the constraints imposed by the MG feature calculus that control the derivation.
- One can now add features to subtrees whose features have already been discharged and thus reactivate them, and there is no bound on how often this can be done for any given subtree.

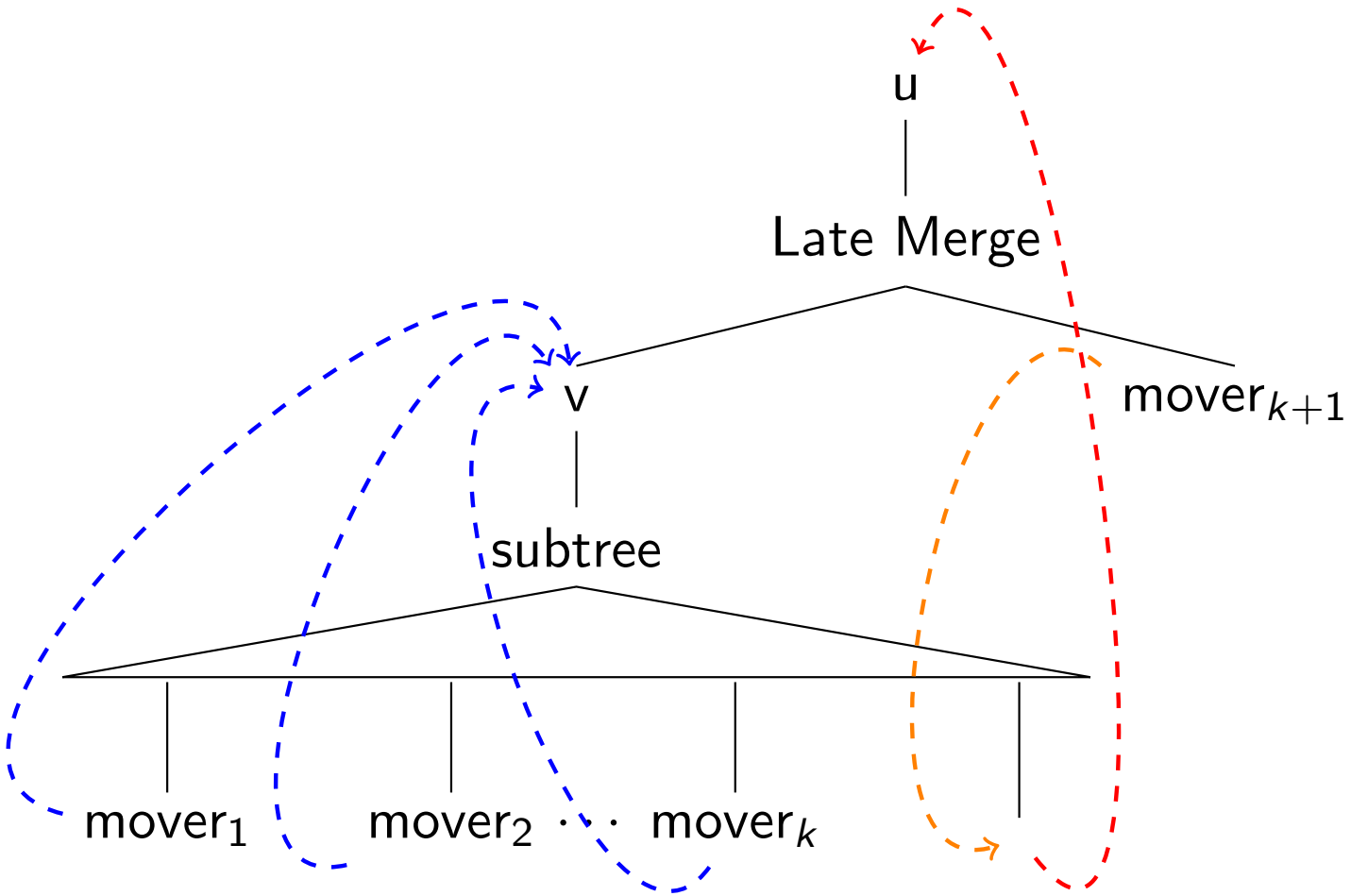
Example of Feature "Smuggling" via Late Merge

Derivation below has more than k parallel movers



Example of Feature "Smuggling" via Late Merge [cont.]

Strongly equivalent derivation with no more than k parallel movers



Is Late Merge Necessary?

Playing Devil's Advocate

- The arguments for Late Merge are very theory specific.
- The exceptions can be coded directly into the constraints and operations.
- Extra power of Late Merge goes unused.
- Late Merge makes the formalism needlessly more complicated.

The Crucial Point

- While Late Merge is not needed to generate the right structures, it generalizes across a variety of domains.
- Grammar formalisms need not only generate the right output, they also need to be able to **express generalizations**.
- So let's see if Late Merge can be implemented in a more restrictive manner.

Is Late Merge Necessary?

Playing Devil's Advocate

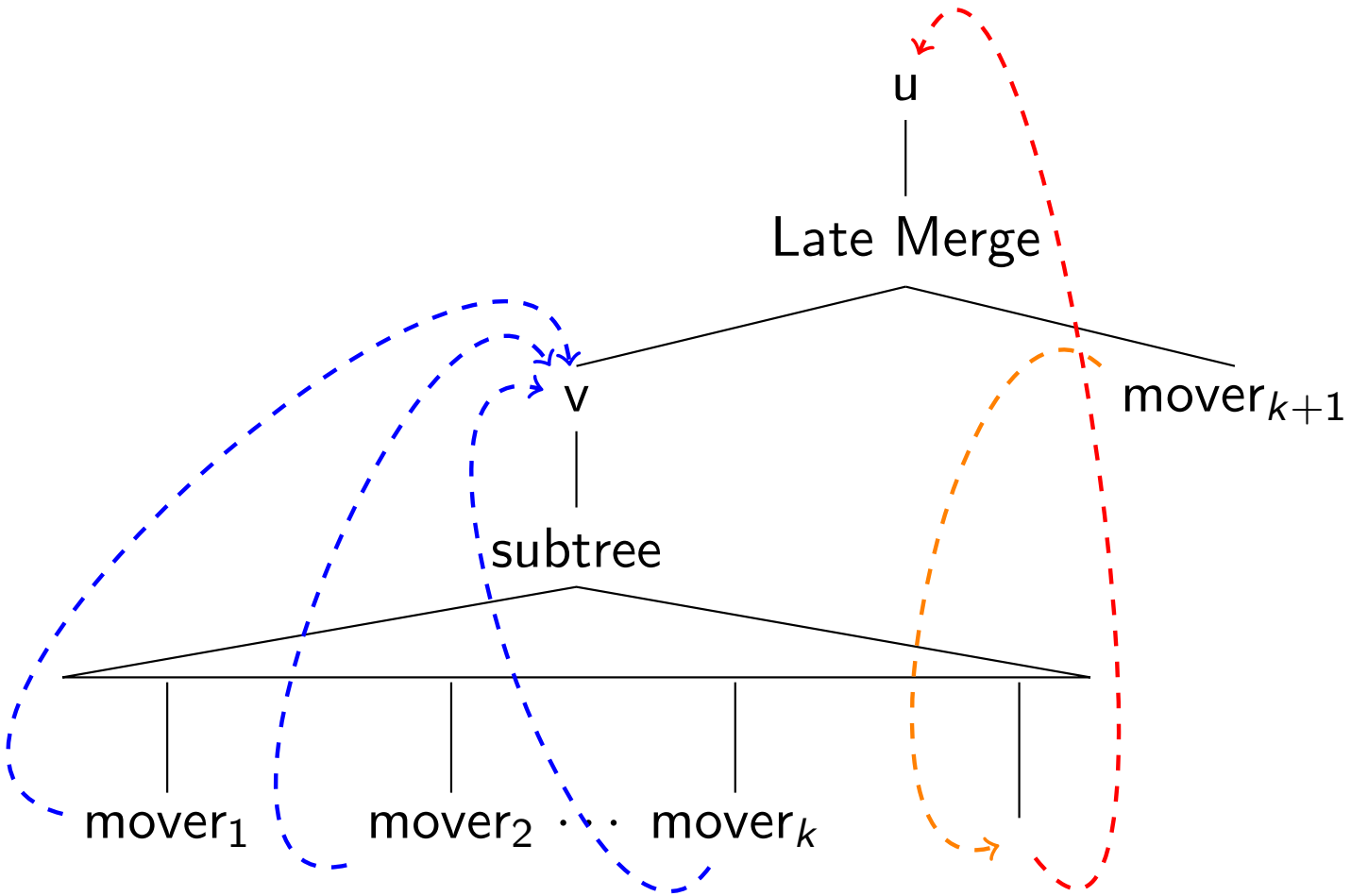
- The arguments for Late Merge are very theory specific.
- The exceptions can be coded directly into the constraints and operations.
- Extra power of Late Merge goes unused.
- Late Merge makes the formalism needlessly more complicated.

The Crucial Point

- While Late Merge is not needed to generate the right structures, it generalizes across a variety of domains.
- Grammar formalisms need not only generate the right output, they also need to be able to **express generalizations**.
- So let's see if Late Merge can be implemented in a more restrictive manner.

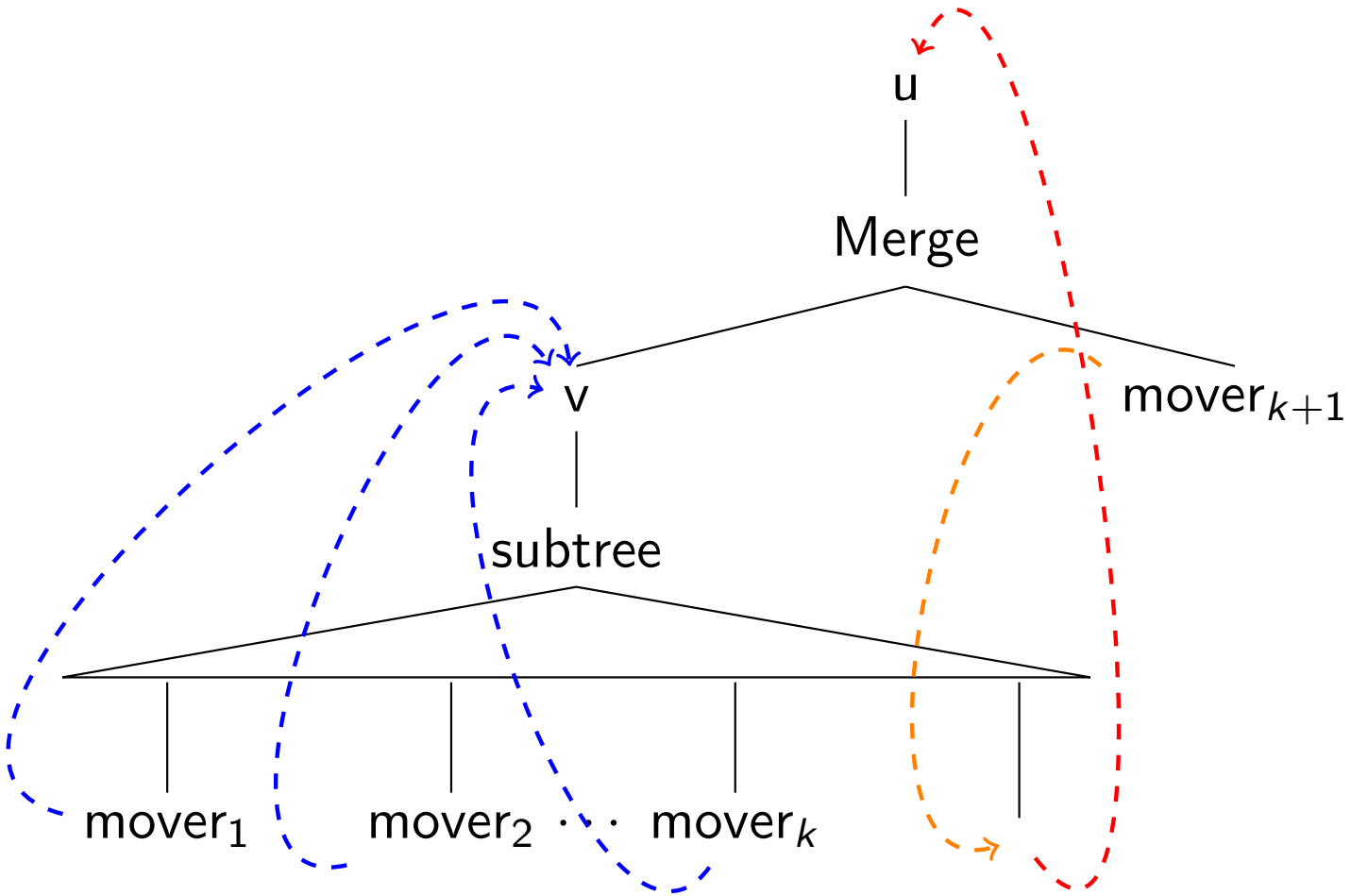
Late Merge as Lowering

The Late Merge derivation can be viewed as **standard Merge followed by Lowering**.



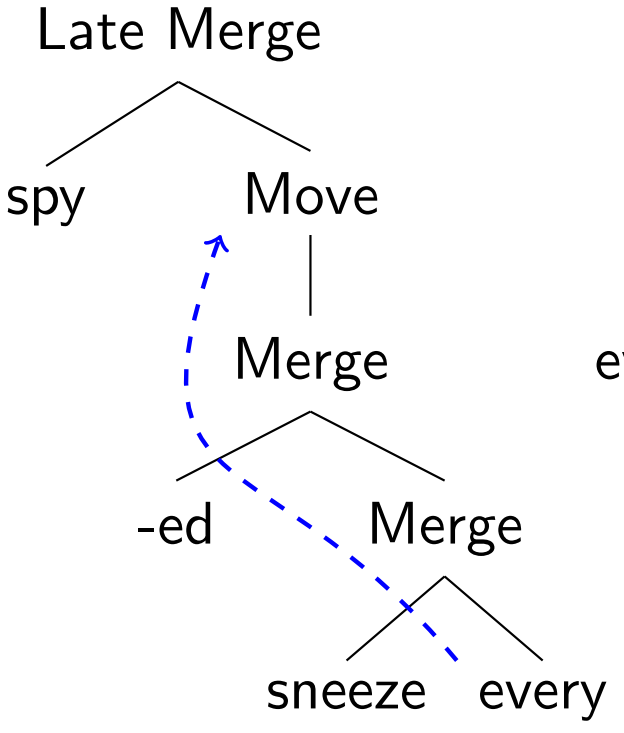
Late Merge as Lowering

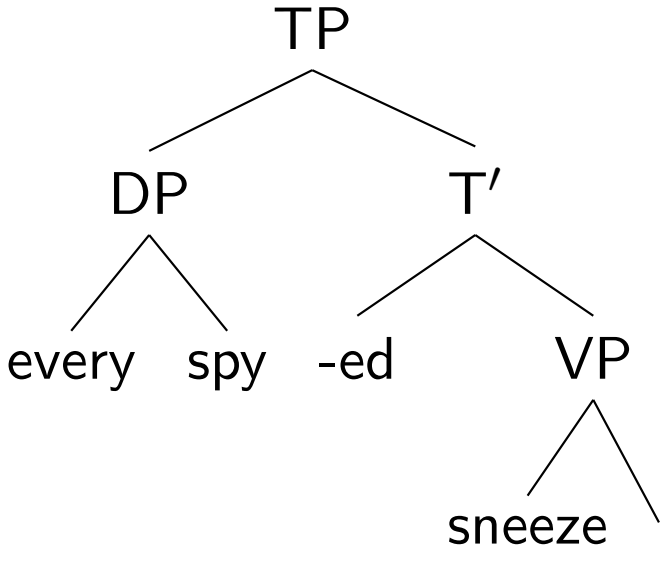
The Late Merge derivation can be viewed as **standard Merge followed by Lowering**.



Late Merger of Argument in Minimalist Literature

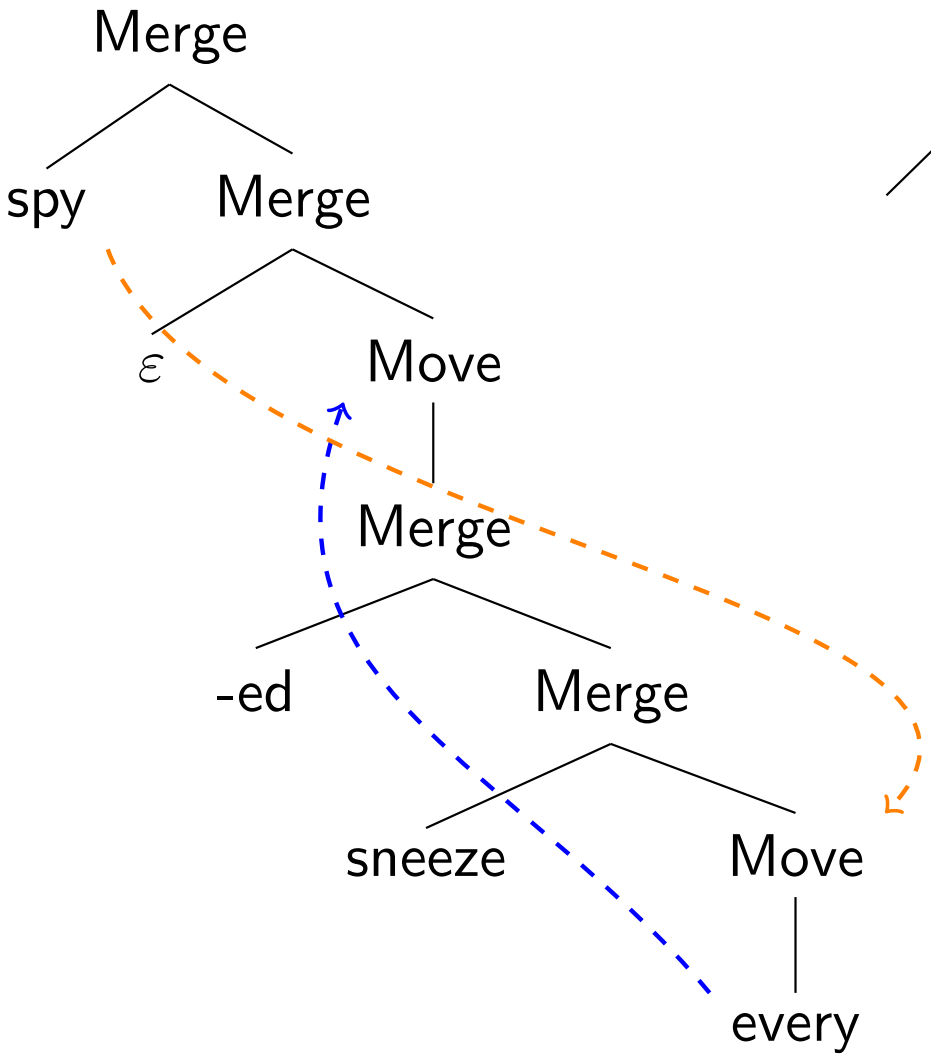
- -ed sneeze [$_{DP}$ every]
- [$_{DP}$ every] -ed sneeze
- [$_{DP}$ every spy] -ed sneeze

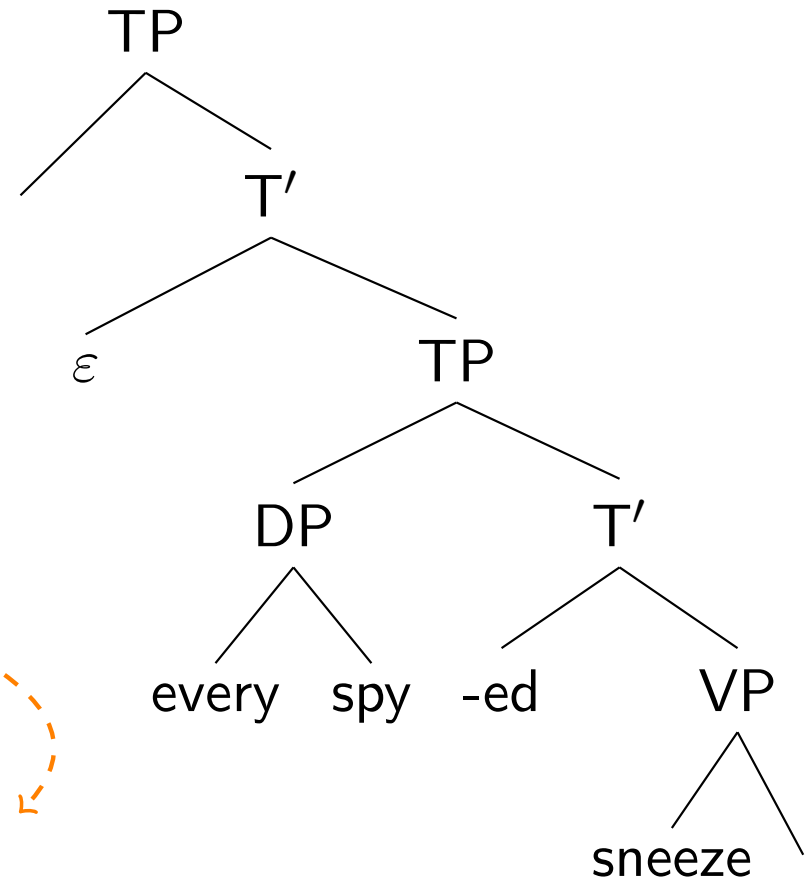




Late Merger of Argument via Lowering

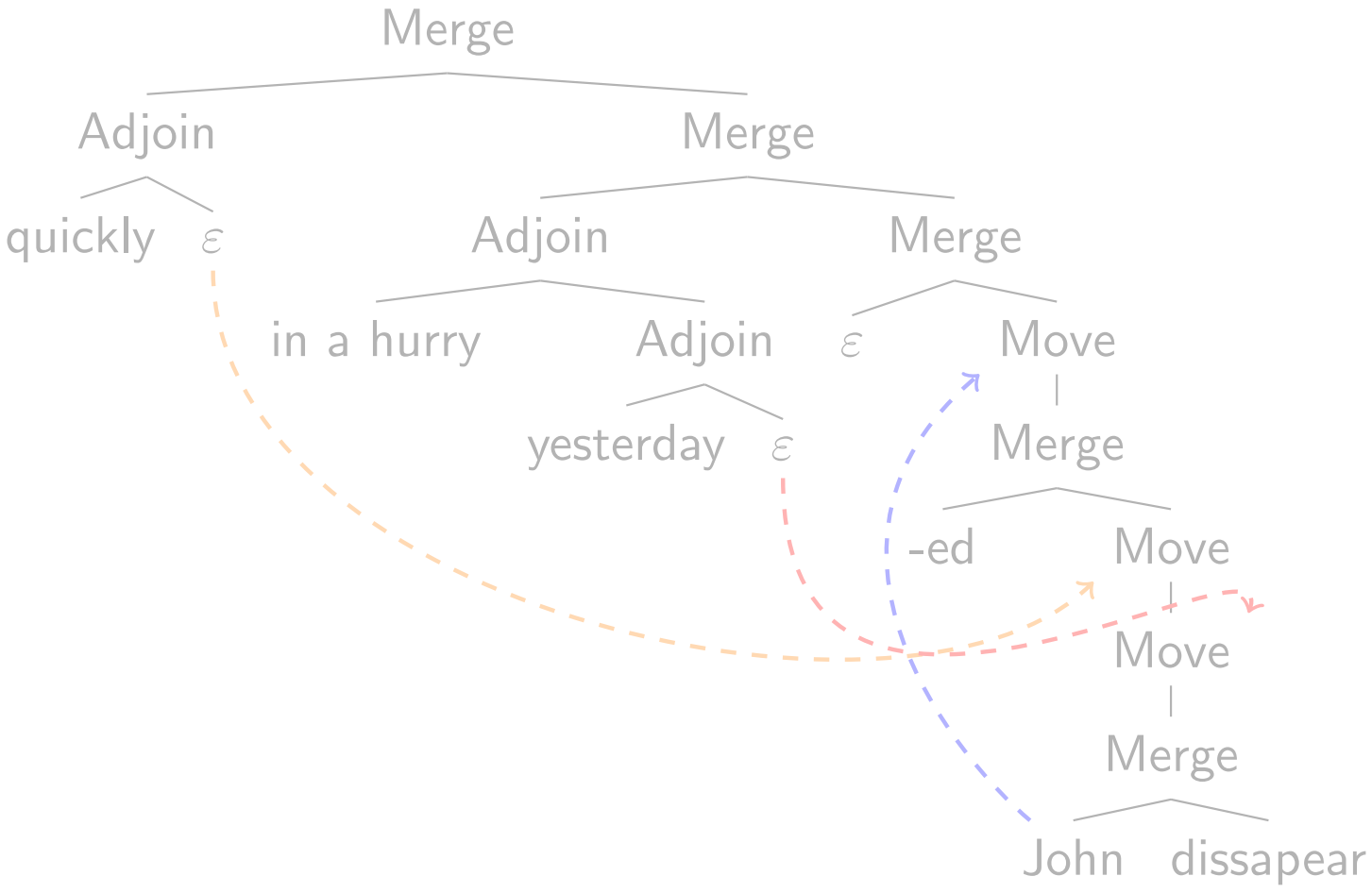
Lowering produces the same output structure (modulo empty head)





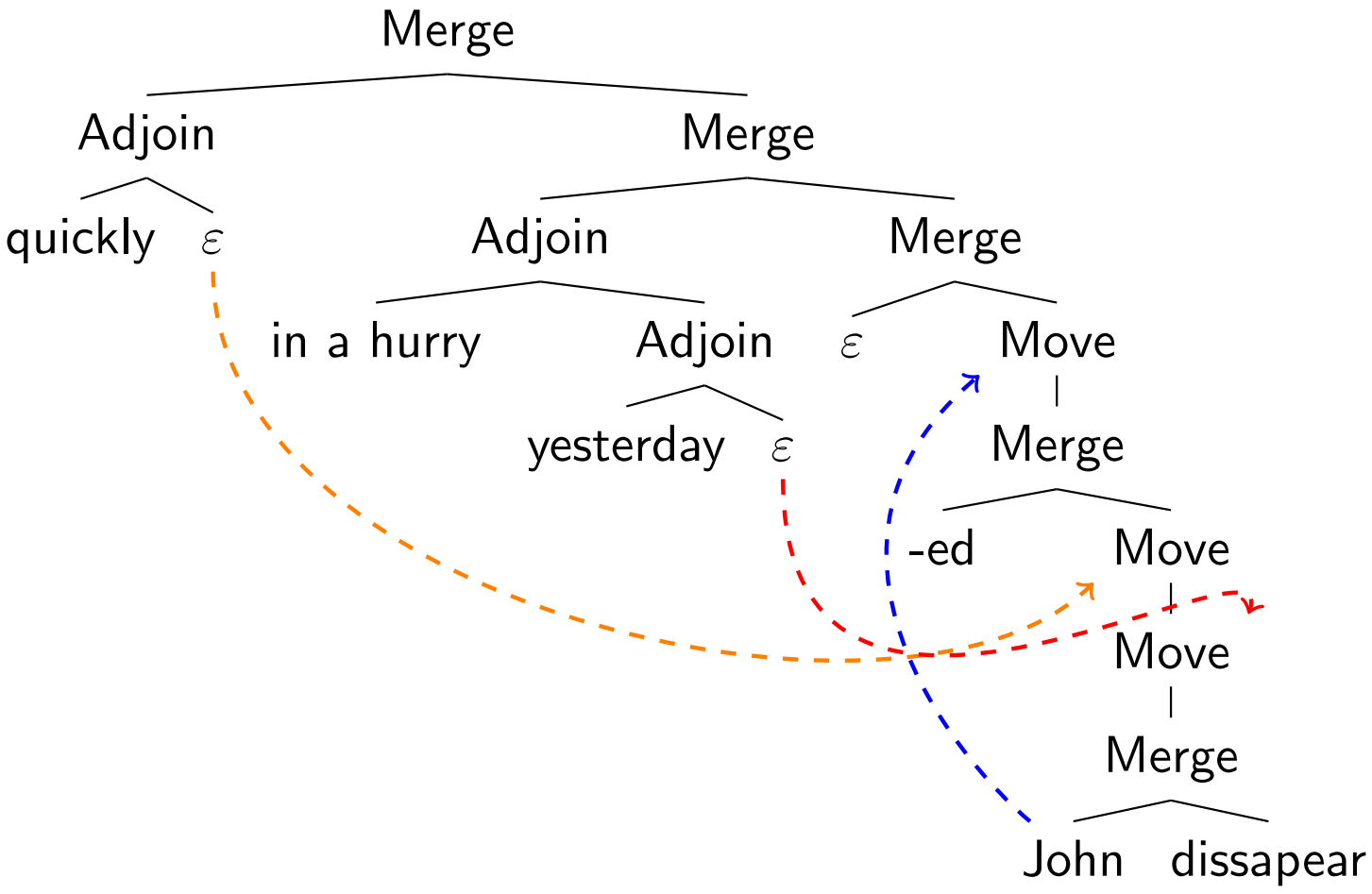
A Minor Problem with Adjuncts

- No upper bound on the number of adjuncts per phrase
- Only k-many phrases may be lowered at the same time
- **Solution:** Piggy-backing!

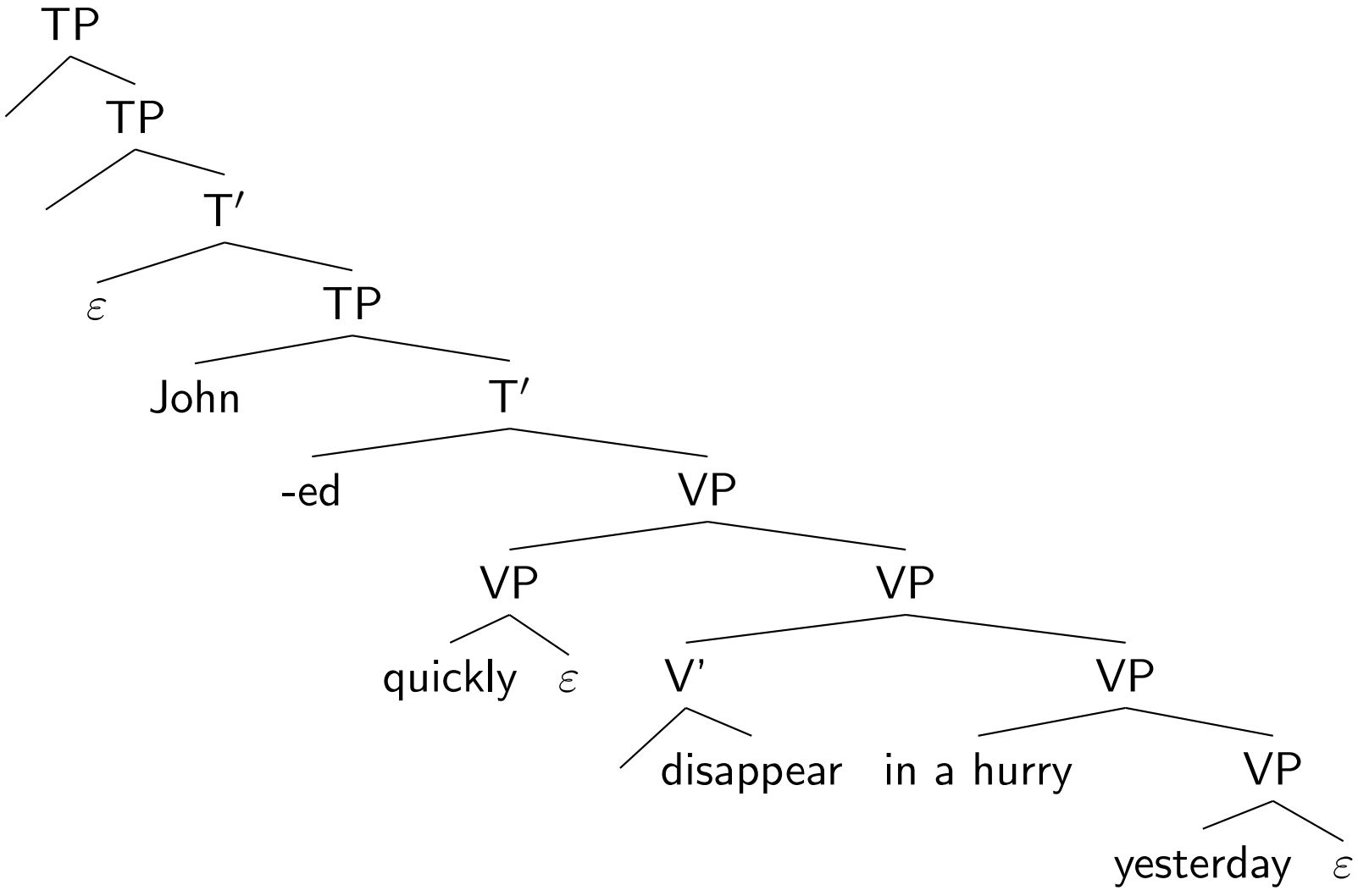


A Minor Problem with Adjuncts

- No upper bound on the number of adjuncts per phrase
- Only k-many phrases may be lowered at the same time
- **Solution:** Piggy-backing!



Output Structure via Piggy-Backing



Some Linguistic Comments on Piggy-Backing

Syntactic Validity

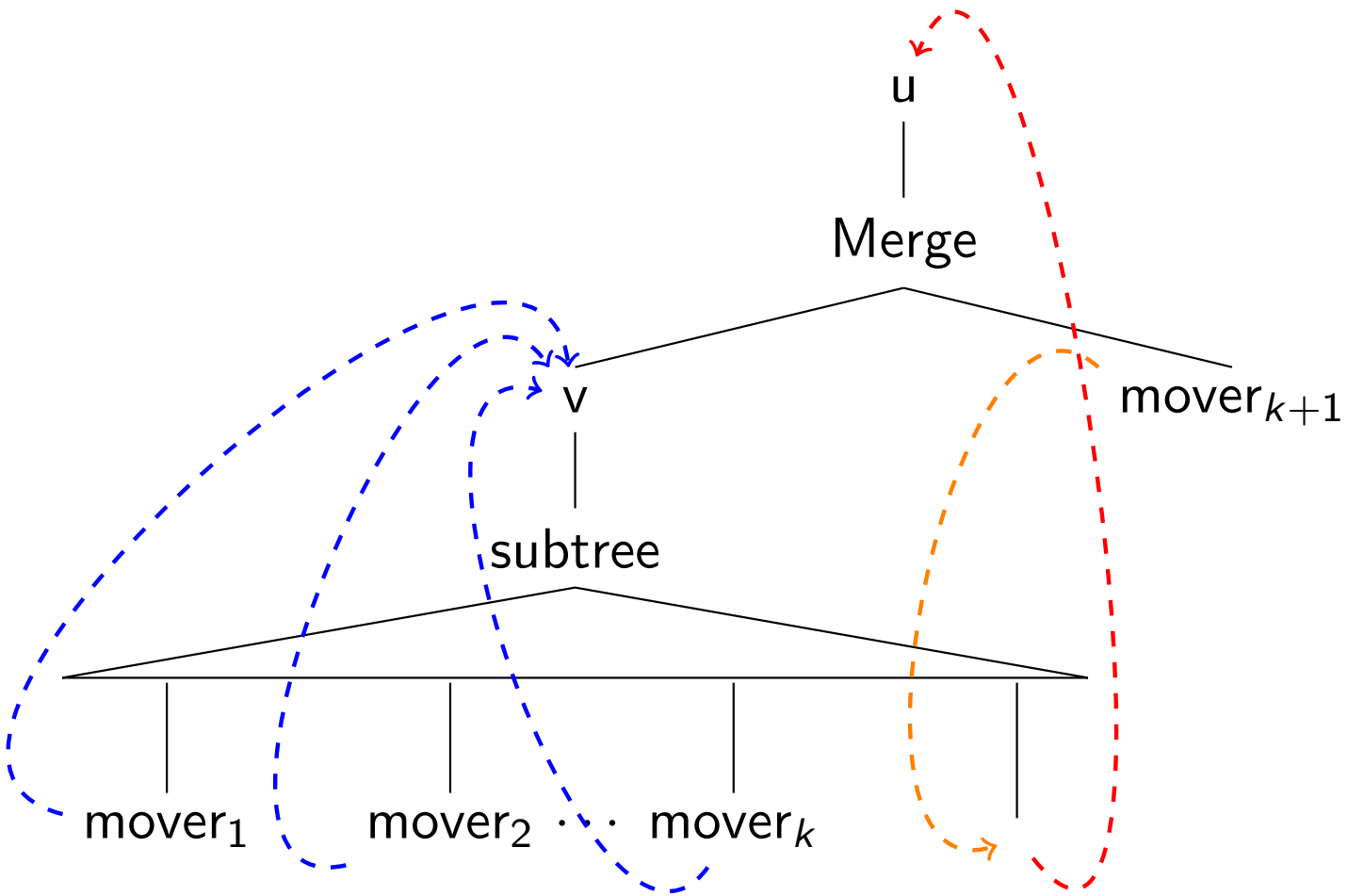
- Piggy-backed VP-adjuncts no longer c-command the VP, but this is never needed anyways.
- A third empty head may be added in order to get the right constituency for partial VP-ellipsis.

Semantic Validity

- Adjuncts can keep standard meaning
- Empty heads undergoing lowering denote some higher-order function that combines the meaning of the adjuncts with the meaning of the VP.

The Limits of Lowering: Feature Smuggling is Impossible

If MG G has at most k parallel movers, the derivation below is blocked because Lowering behaves like **derivational time travel**.



Formal Evaluation of Lowering Implementation

- **Linguistically adequate**
  - Late Merge of arguments straight-forward
  - Late Merge of adjuncts doable with piggy-backing
- **Weaker than Late Merge**
  - MGMGs evaluate the upper limit on movers in a global fashion.
  - Hence lowering cannot smuggle in new movers.
  - Weak generative capacity of MGMGs is preserved.
- **Preserves strong generative capacity of MGs**
  - Lowering increases power of MGs only if it can follow raising.
  - Lowering as Late Merge is always the first movement step.

What Have We Learned?

- Late Merge is invoked by Minimalists for several phenomena.
- As specified in the **literature**, it is a very powerful operation that **pushes MGs beyond MCFLs**.
- As used by **linguists**, it is easily **emulated by lowering**.
- Neither weak nor strong generative capacity of MGs are increased (but complexity of mapping to derived trees is).

The Big Picture

- Late Merge is just one of many operations that is not part of the MG-toolbox.
  - Sidewards movement
  - Affix hopping
  - TAG-style adjunction
- All of them can be captured by adding lowering.
  - Every MGMG movement-type can be decomposed into at most one raising step followed by at most one lowering step.
  - Late Merge is actually one of the simplest operations.
- Raising and lowering are subtypes of MSO transductions.

Conjecture: composition of these two types yields whole class of MSO transductions.

References I

Chomsky, Noam. 1995. *The minimalist program*. Cambridge, Mass.: MIT Press.

Gärtner, Hans-Martin, and Jens Michaelis. 2008. A note on countercyclicity and minimalist grammars. In *Proceedings of Formal Grammar 2003*, ed. Gerald Penn, 95–109. Stanford: CSLI-Online.

Graf, Thomas. 2012a. Locality and the complexity of minimalist derivation tree languages. In *Formal Grammar 2010/2011*, ed. Philippe de Groot and Mark-Jan Nederhof, volume 7395 of *Lecture Notes in Computer Science*, 208–227. Heidelberg: Springer.

Graf, Thomas. 2012b. Movement-generalized minimalist grammars. In *LACL 2012*, ed. Denis Béchet and Alexander J. Dikovsky, volume 7351 of *Lecture Notes in Computer Science*, 58–73.

Graf, Thomas. 2012c. Tree adjunction as minimalist lowering. In *Proceedings of the 11th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+11)*, 19–27.

Graf, Thomas. 2013. *Local and transderivational constraints in syntax and semantics*. Doctoral Dissertation, UCLA.

References II

Harkema, Henk. 2001. A characterization of minimalist languages. In *Logical aspects of computational linguistics (LACL'01)*, ed. Philippe de Groote, Glyn Morrill, and Christian Retoré, volume 2099 of *Lecture Notes in Artificial Intelligence*, 193–211. Berlin: Springer.

Kobele, Gregory M. 2010. On late adjunction in minimalist grammars. Slides for a talk given at MCFG+ 2010.

Kobele, Gregory M., and Jens Michaelis. 2011. Disentangling notions of specifier impenetrability. In *The Mathematics of Language*, ed. Makoto Kanazawa, András Kornia, Marcus Kracht, and Hiroyuki Seki, volume 6878 of *Lecture Notes in Artificial Intelligence*, 126–142.

Kobele, Gregory M., Christian Retoré, and Sylvain Salvati. 2007. An automata-theoretic approach to minimalism. In *Model Theoretic Syntax at 10*, ed. James Rogers and Stephan Kepser, 71–80.

Lebeaux, David. 1988. *Language acquisition and the form of the grammar*. Doctoral Dissertation, University of Massachusetts, Amherst.

Michaelis, Jens. 2001. Transforming linear context-free rewriting systems into minimalist grammars. *Lecture Notes in Artificial Intelligence* 2099:228–244.

Mönnich, Uwe. 2006. Grammar morphisms. Ms. University of Tübingen.

References III

Mönnich, Uwe. 2007. Minimalist syntax, multiple regular tree grammars and direction preserving tree transductions. In *Model Theoretic Syntax at 10*, ed. James Rogers and Stephan Kepser, 83–87.

Morawietz, Frank. 2003. *Two-step approaches to natural language formalisms*. Berlin: Walter de Gruyter.

Ochi, Masao. 1999. Multiple spell-out and PF adjacency. In *Proceedings of the North Eastern Linguistic Society*, volume 29.

Stabler, Edward P. 1997. Derivational minimalism. In *Logical aspects of computational linguistics*, ed. Christian Retoré, volume 1328 of *Lecture Notes in Computer Science*, 68–95. Berlin: Springer.

Stepanov, Arthur. 2001. Late adjunction and minimalist phrase structure. *Syntax* 4:94–125.

Takahashi, Shoichi, and Sarah Hulsey. 2009. Wholesale Late Merger: Beyond the A/$\overline{\text{A}}$ distinction. *Linguistc Inquiry* 40:387–426.