Commonality in Disparity: The Computational View of Syntax and Phonology

Thomas Graf
Stony Brook University
mail@thomasgraf.net
http://thomasgraf.net

Jeffrey Heinz
University of Delaware
heinz@udel.edu
http://udel.edu/~heinz

Glow 38
April 18 2015

A New View of the Power of Syntax and Phonology

Computations can be factored into two components:

Data Structure	strings, trees, graphs, ...
Algorithm	mechanism for manipulating data structures

Standard Perspective: Weak Generative Capacity

	Phonology	Syntax
Data Structure	string	string
Algorithm	regular	**beyond regular**
	(Kaplan and Kay 1994)	(Shieber 1985)

Our Perspective: Subregular Hypothesis

	Phonology	Syntax
Data Structure	string	**trees**
Algorithm	tier-based SL	tier-based SL
	(Heinz et al. 2011)	(this talk)

A New View of the Power of Syntax and Phonology

Computations can be factored into two components:

Data Structure	strings, trees, graphs, . . .
Algorithm	mechanism for manipulating data structures

Standard Perspective: Weak Generative Capacity

	Phonology	**Syntax**
Data Structure	string	string
Algorithm	regular	**beyond regular**
	(Kaplan and Kay 1994)	(Shieber 1985)

Our Perspective: Subregular Hypothesis

	Phonology	**Syntax**
Data Structure	string	**trees**
Algorithm	tier-based SL	tier-based SL
	(Heinz et al. 2011)	(this talk)

Outline

1. Phonology is Tier-Based Strictly Local
2. Minimalist Grammars as a Formal Model of Syntax
3. Syntax is Tier-Based Strictly Local
    - Tree *n*-gram Grammars
    - Regulating Movement via Tree-Tiers

Phonology as a Formal Language

Like the standard perspective, we view phonology as a set of well-formed strings $\Rightarrow$ **phonology** $\equiv$ **phonotactics**

Subregular Hypothesis (Weak Version; Heinz et al. 2011)

Phonology is properly included in the class of regular languages:

- All local dependencies can be described by *n*-gram grammars.
- Non-local dependencies are **local on phonological tiers**.

Remarks

- Primary stress in Creek and Cairene Arabic might not be tier-local, but the data is unclear. (Graf 2010)
- Non-local dependencies might be even weaker. (Heinz 2010)
- The subregular hypothesis might even apply to input-output mappings. (Chandlee 2014)

Phonology as a Formal Language

Like the standard perspective, we view phonology as a set of well-formed strings $\Rightarrow$ **phonology** $\equiv$ **phonotactics**

Subregular Hypothesis (Weak Version; Heinz et al. 2011)

Phonology is properly included in the class of regular languages:

- All local dependencies can be described by *n*-gram grammars.
- Non-local dependencies are **local on phonological tiers**.

Remarks

- Primary stress in Creek and Cairene Arabic might not be tier-local, but the data is unclear. (Graf 2010)
- Non-local dependencies might be even weaker. (Heinz 2010)
- The subregular hypothesis might even apply to input-output mappings. (Chandlee 2014)

(Negative) Bigram Grammars for Local Processes

- Suppose we have a fixed alphabet Σ (e.g. sounds of English).
- A bigram is a sequence **ab** s.t. **a** and **b** are members of Σ.
- A **bigram grammar** G is a finite set of bigrams.
- G generates the largest language of strings such that no string contains any bigrams of G as a substring
- Intuition: **bigrams are hard, local constraints**

Example

Rewrite rule	Constraint	Bigrams
n $\rightarrow$ m $\vert$ _ **b**	***nb**	**nb**
z $\rightarrow$ s $\vert$ _ **\$**	***z\$**	**z\$**
[-voice] $\rightarrow$ $\emptyset$ $\vert$ _ **\$**	***[-voice]\$**	**s\$**, **θ\$**, **f\$**, ...

(Negative) Bigram Grammars for Local Processes

- Suppose we have a fixed alphabet Σ (e.g. sounds of English).
- A bigram is a sequence **ab** s.t. **a** and **b** are members of Σ.
- A **bigram grammar** G is a finite set of bigrams.
- G generates the largest language of strings such that no string contains any bigrams of G as a substring
- Intuition: **bigrams are hard, local constraints**

Example

Rewrite rule	Constraint	Bigrams
n $\rightarrow$ m $\mid$ _ **b**	***nb**	**nb**
z $\rightarrow$ s $\mid$ _ **$**	***z$**	**z$**
[-voice] $\rightarrow$ $\emptyset$ $\mid$ _ **$**	***[-voice]$**	**s$**, **θ$**, **f$**, ...

Tiers for Long-Distance Dependencies

- We can move to 3-grams, 4-grams, ... n-grams in order to regulate less local processes (e.g. umlaut, vowel harmony).
- **Problem:** Still limited to locality domain of size n
  $\Rightarrow$ unbounded processes cannot be captured
- **Solution:** segments can be on multiple tiers

Tier-Based Bigram Grammar

- Tier-projection is determined by the shape of the segment, not by structural properties (e.g. feet).
- A string is well-formed iff no tier T contains an illicit T-n-gram.

Tiers for Long-Distance Dependencies

- We can move to 3-grams, 4-grams, . . . n-grams in order to regulate less local processes (e.g. umlaut, vowel harmony).
- **Problem:** Still limited to locality domain of size n
  $\Rightarrow$ unbounded processes cannot be captured
- **Solution:** segments can be on multiple tiers

Tier-Based Bigram Grammar

- Tier-projection is determined by the shape of the segment, not by structural properties (e.g. feet).
- A string is well-formed iff no tier T contains an illicit T-n-gram.

Example: Sibilant Harmony

Rewrite rule	Constraint	Tier_1-Bigram
s → ʃ | ʃ ··· _	*ʃ ··· s	ʃs

Tier_1 contains all sibilants
Tier_0 contains all segments

Tier_1:	\$		ʃ		s		\$
Tier_0:	\$	e	ʃ	i	s	i	\$

Tier_1:	\$		ʃ		ʃ		\$
Tier_0:	\$	e	ʃ	i	ʃ	i	\$

A Closer Look at Syntax

- Phonology is tier-based strictly local and thus subregular.
- Syntactic dependencies, on the other hand, yield non-regular string sets.
- But: **syntacticians work with trees**, not strings.
- Minimalist grammars (MGs) are a formalization of Minimalist syntax that makes this idea precise. (Stabler 1997)

Syntax as Chemistry of Language

Minimalist grammars treat syntax like chemistry.

Chemistry	**Syntax**
atoms	words
electrons	features
molecules	sentences

- Every word is a collection of features.
- Every feature has either positive or negative polarity.
- Features of opposite polarity annihilate each other.
- Feature annihilation drives the structure-building operations **Merge** and **Move**.
- **Merge and Move do all the work**, there are no other mechanisms like the EPP or the Θ-criterion.

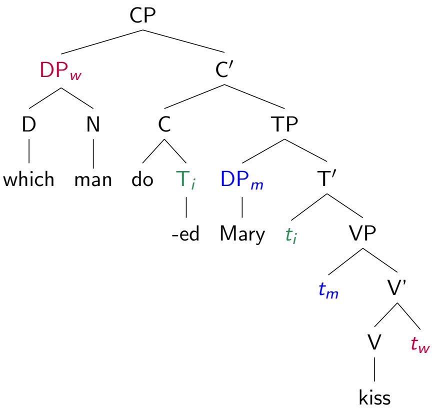
Syntax as Chemistry of Language

Minimalist grammars treat syntax like chemistry.

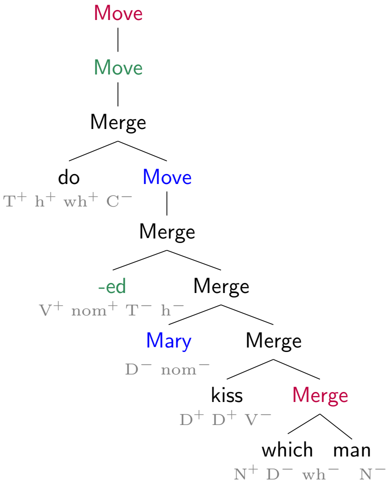
Chemistry	**Syntax**
atoms	words
electrons	features
molecules	sentences

- Every word is a collection of features.
- Every feature has either positive or negative polarity.
- Features of opposite polarity annihilate each other.
- Feature annihilation drives the structure-building operations **Merge** and **Move**.
- **Merge and Move do all the work**, there are no other mechanisms like the EPP or the Θ-criterion.

MG Syntax in Action



Phrase Structure Tree



Derivation Tree

The Complexity of Minimalist Tree Languages

- Syntacticians usually look at the tree structure that is built by the operations Merge and Move.
- But the history of how such a structure is built is also a tree
  $\Rightarrow$ **phrase structure trees** and **derivation trees** as two possible views of tree-based syntax
- The set of phrase structure trees is not regular.
  (Doner 1970; Thatcher 1967; Michaelis 2001)
- But the set of derivation trees is regular.
  (Michaelis 2001; Kobele et al. 2007; Graf 2012)

The Big Question

Are MG derivation tree languages tier-based strictly local?

The Complexity of Minimalist Tree Languages

- Syntacticians usually look at the tree structure that is built by the operations Merge and Move.
- But the history of how such a structure is built is also a tree
  $\Rightarrow$ **phrase structure trees** and **derivation trees** as two possible views of tree-based syntax
- The set of phrase structure trees is not regular.
  (Doner 1970; Thatcher 1967; Michaelis 2001)
- But the set of derivation trees is regular.
  (Michaelis 2001; Kobele et al. 2007; Graf 2012)

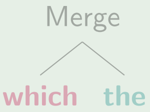
The Big Question

Are MG derivation tree languages tier-based strictly local?

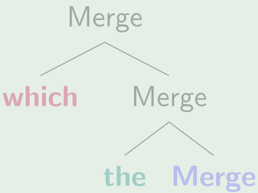
Tree *n*-gram Grammars

- We need to lift *n*-grams from strings to trees.
- Instead of strings of length *n*, use subtrees of depth *n*.
- Each subtree encodes a constraint on the derivation.

Example: Regulating Illicit Merge



block Merger of **which** and **the**

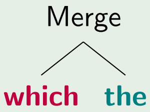


block Merger of **which** and **the XP**

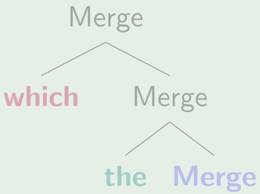
Tree *n*-gram Grammars

- We need to lift *n*-grams from strings to trees.
- Instead of strings of length n, use subtrees of depth n.
- Each subtree encodes a constraint on the derivation.

Example: Regulating Illicit Merge



block Merger of **which** and **the**

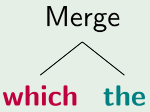


block Merger of **which** and **the XP**

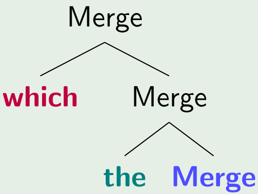
Tree *n*-gram Grammars

- We need to lift *n*-grams from strings to trees.
- Instead of strings of length *n*, use subtrees of depth *n*.
- Each subtree encodes a constraint on the derivation.

Example: Regulating Illicit Merge



block Merger of **which** and **the**



block Merger of **which** and **the XP**

Constraints on Move

Merge is a local process, regulated by tree *n*-grams.
But what about Move?

Suppose our MG is in **single movement normal form**,
i.e. every lexical item moves at most once.
Then movement is regulated by two constraints. (Graf 2012)

Constraints on Movement

Move	Every lexical item with a negative Move feature has a dominating matching Move node.
SMC	Every Move node is a closest dominating match for exactly one lexical item.

Constraints on Move

Merge is a local process, regulated by tree *n*-grams.
But what about Move?

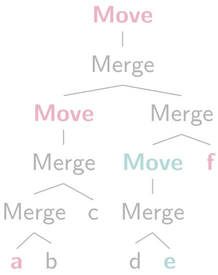
Suppose our MG is in **single movement normal form**,
i.e. every lexical item moves at most once.
Then movement is regulated by two constraints. (Graf 2012)

Constraints on Movement

Move	Every lexical item with a negative Move feature has a dominating matching Move node.
SMC	Every Move node is a closest dominating match for exactly one lexical item.

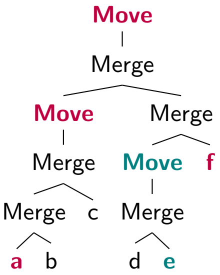
Tiers for Movement

- There is no upper bound on the distance between a lexical item and its matching Move node.
- Consequently, **Move dependencies are not strictly local**.
- What if every movement type (wh, topic, . . .) induces its own tier? Would that make Move dependencies local?



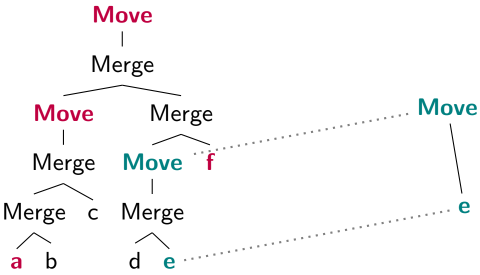
Tiers for Movement

- There is no upper bound on the distance between a lexical item and its matching Move node.
- Consequently, **Move dependencies are not strictly local**.
- What if every movement type (wh, topic, . . .) induces its own tier? Would that make Move dependencies local?



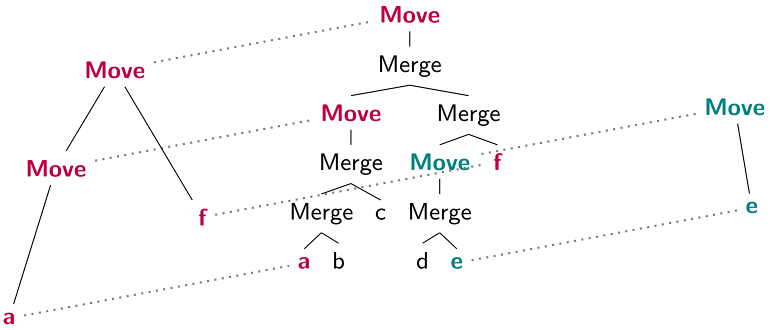
Tiers for Movement

- There is no upper bound on the distance between a lexical item and its matching Move node.
- Consequently, **Move dependencies are not strictly local**.
- What if every movement type (wh, topic, . . .) induces its own tier? Would that make Move dependencies local?



Tiers for Movement

- There is no upper bound on the distance between a lexical item and its matching Move node.
- Consequently, **Move dependencies are not strictly local**.
- What if every movement type (wh, topic, . . .) induces its own tier? Would that make Move dependencies local?



Move Constraints over Tiers

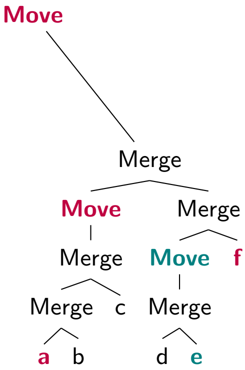
	Original	**Tier**
Move	Every lexical item with a negative Move feature has a dominating matching Move node.	Every lexical item has a **mother** labeled Move.
SMC	Every Move node is a closest dominating match for exactly one lexical item.	Exactly one of a Move node's **daughters** is a lexical item.

Tree Bigram Templates

Move	**SMC1**	**SMC2**
\$	Move	Move
LI	no LI	$\geq$ 2 LIs

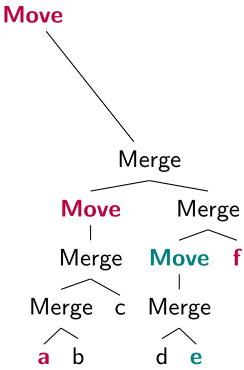
The Problem With Our Bigrams

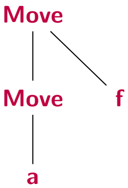
- No limit on number of daughters per Move node in tier
  $\Rightarrow$ **SMC1** and **SMC2** correspond to infinitely many bigrams
- But a bigram grammar must be finite!



The Problem With Our Bigrams

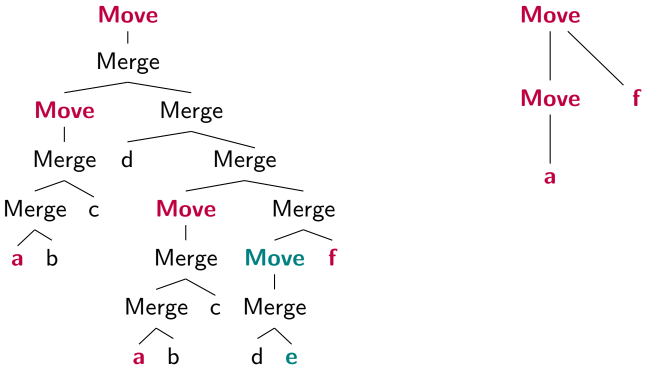
- No limit on number of daughters per Move node in tier
  $\Rightarrow$ **SMC1** and **SMC2** correspond to infinitely many bigrams
- But a bigram grammar must be finite!





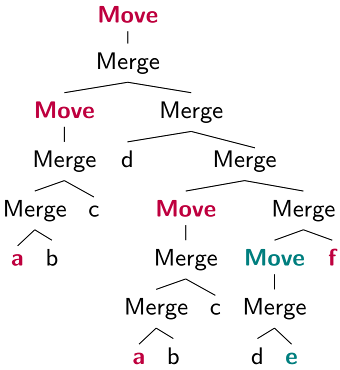
The Problem With Our Bigrams

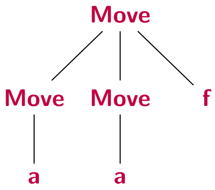
- No limit on number of daughters per Move node in tier
  $\Rightarrow$ **SMC1** and **SMC2** correspond to infinitely many bigrams
- But a bigram grammar must be finite!



The Problem With Our Bigrams

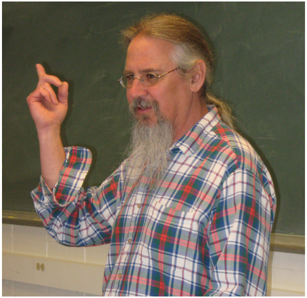
- No limit on number of daughters per Move node in tier
  $\Rightarrow$ **SMC1** and **SMC2** correspond to infinitely many bigrams
- But a bigram grammar must be finite!





A Hint From Multidimensional Trees

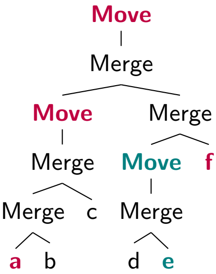
- We think of trees as nodes ordered by dominance and precedence.
- Jim Rogers (2003) formalizes trees as strings (sequences of siblings) related by dominance.
- Analogously, a tree-tier consists of **string-tiers related by dominance**!



Tree-Tiers with Lexical Tiers

Construction Procedure

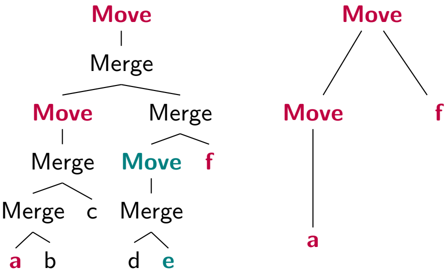
- Take derivation and project Move tiers.
- In every Move tier, project LI-tiers.



Tree-Tiers with Lexical Tiers

Construction Procedure

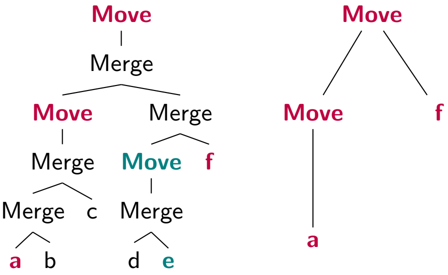
- Take derivation and project Move tiers.
- In every Move tier, project LI-tiers.



Tree-Tiers with Lexical Tiers

Construction Procedure

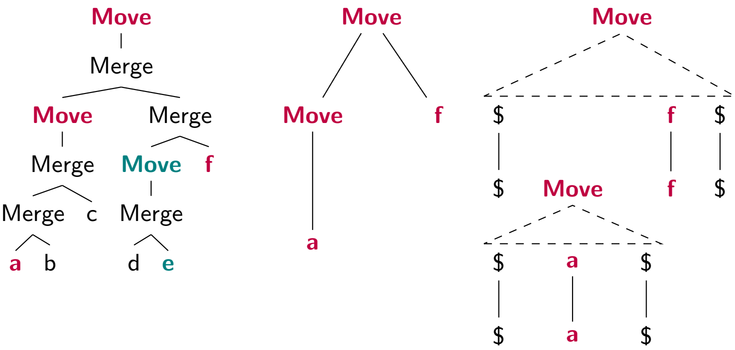
- Take derivation and project Move tiers.
- In every Move tier, project LI-tiers.



Tree-Tiers with Lexical Tiers

Construction Procedure

- Take derivation and project Move tiers.
- In every Move tier, project LI-tiers.



From Templates to Tree Bigrams with Lexical Tiers

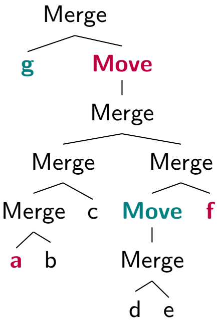
Old Tree Bigram Templates

Move	**SMC1**	**SMC2**
\$	Move	Move
LI	no LI	$\geq$ 2 LIs

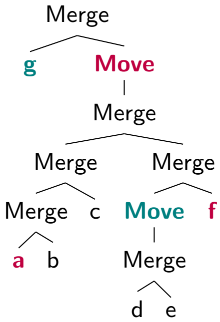
New Tree Bigrams with Lexical Tier *n*-Grams as Daughters

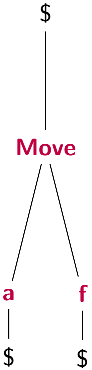
Move	**SMC1**	**SMC2**
\$	Move	Move
LI	\$ \$	LI LI

Example of Ill-Formed Derivation

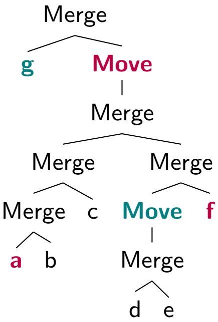


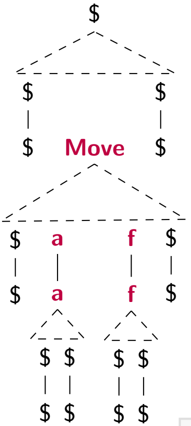
Example of Ill-Formed Derivation



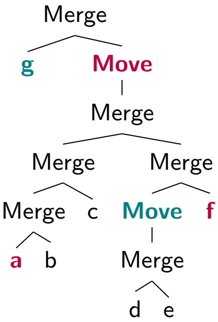


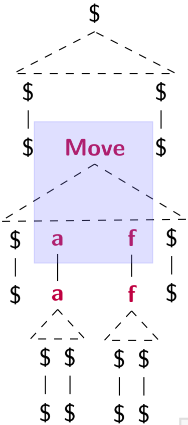
Example of Ill-Formed Derivation



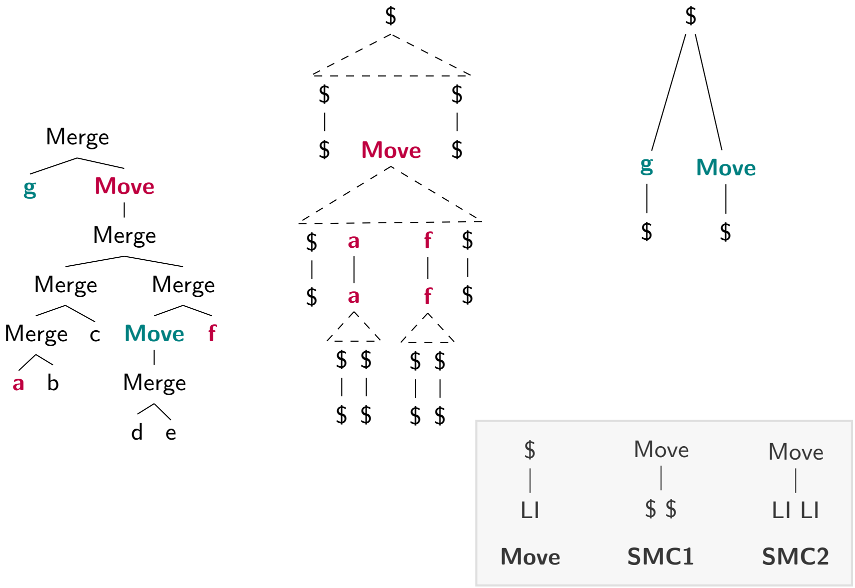


Example of Ill-Formed Derivation

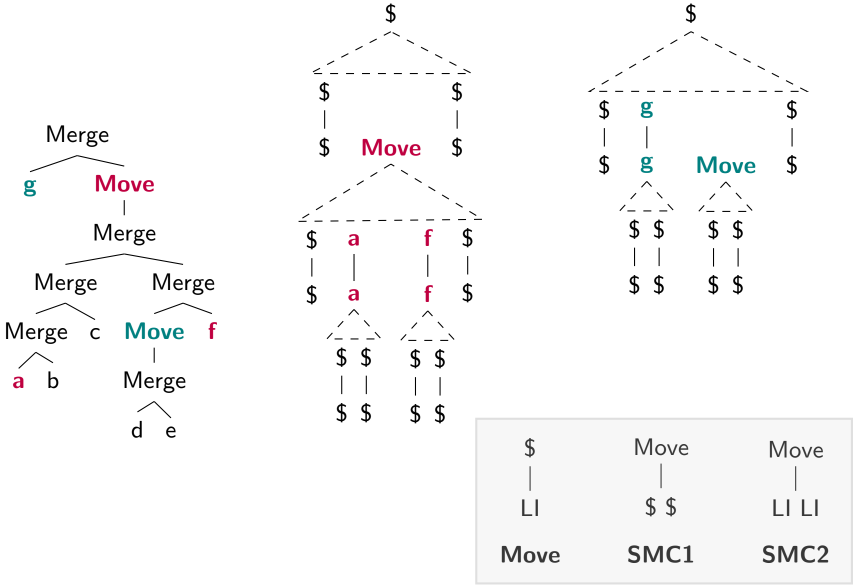




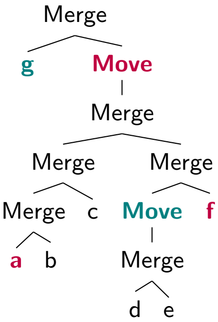
Example of Ill-Formed Derivation

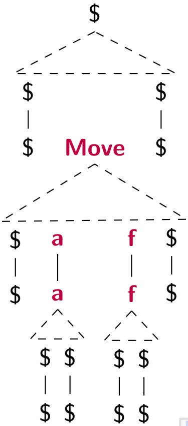


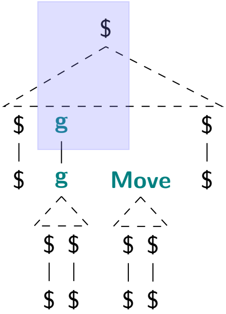
Example of Ill-Formed Derivation



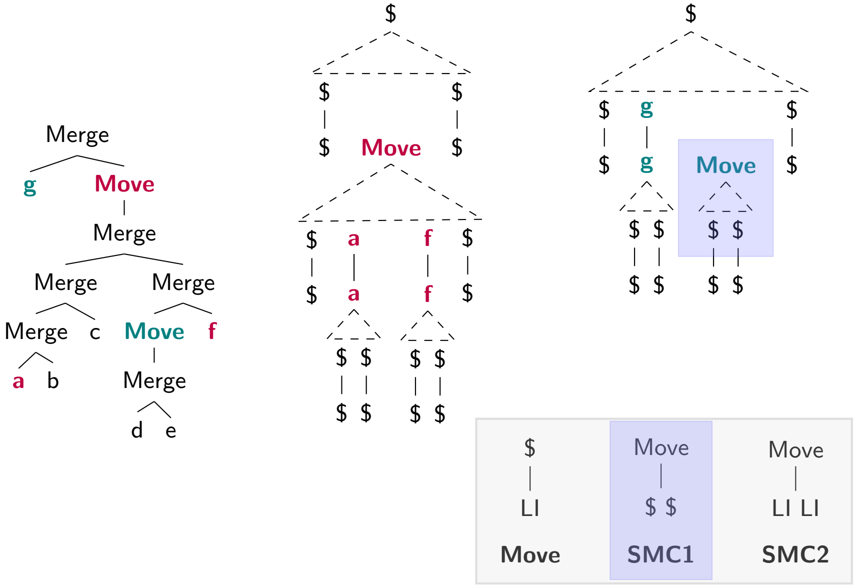
Example of Ill-Formed Derivation



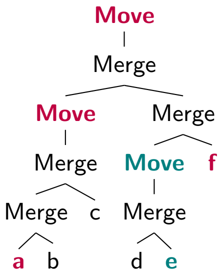




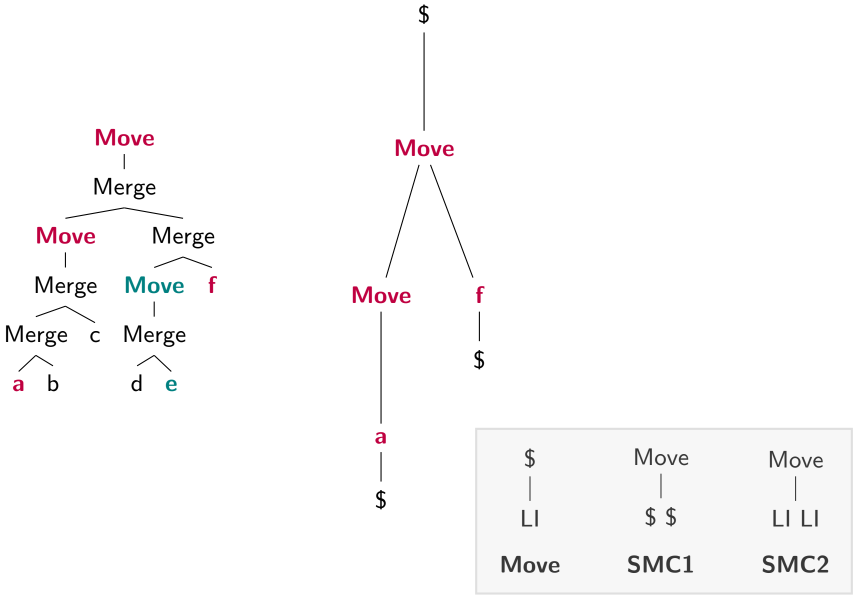
Example of Ill-Formed Derivation



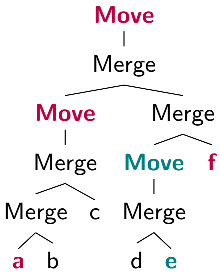
Example of Well-Formed Derivation

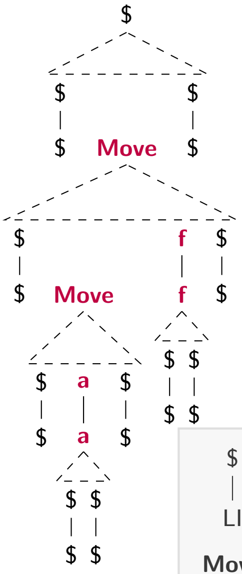


Example of Well-Formed Derivation

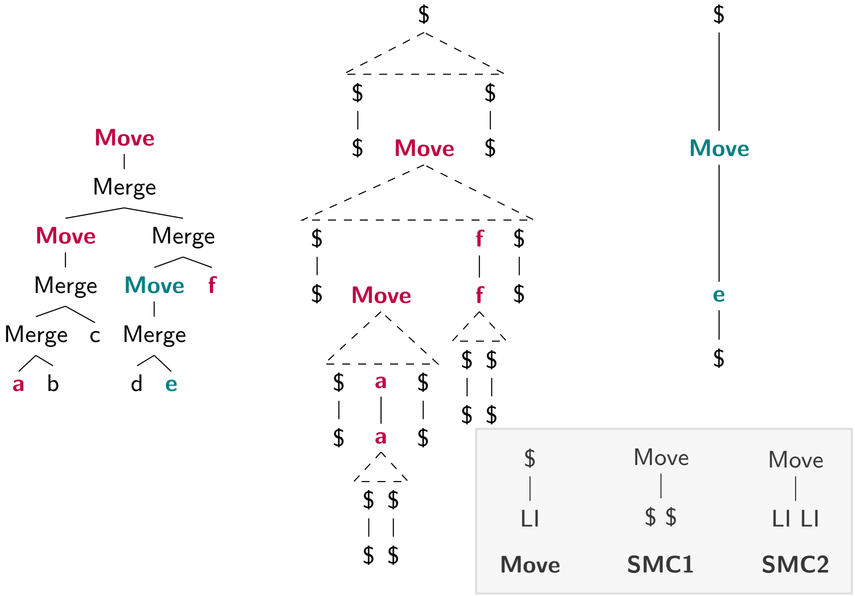


Example of Well-Formed Derivation

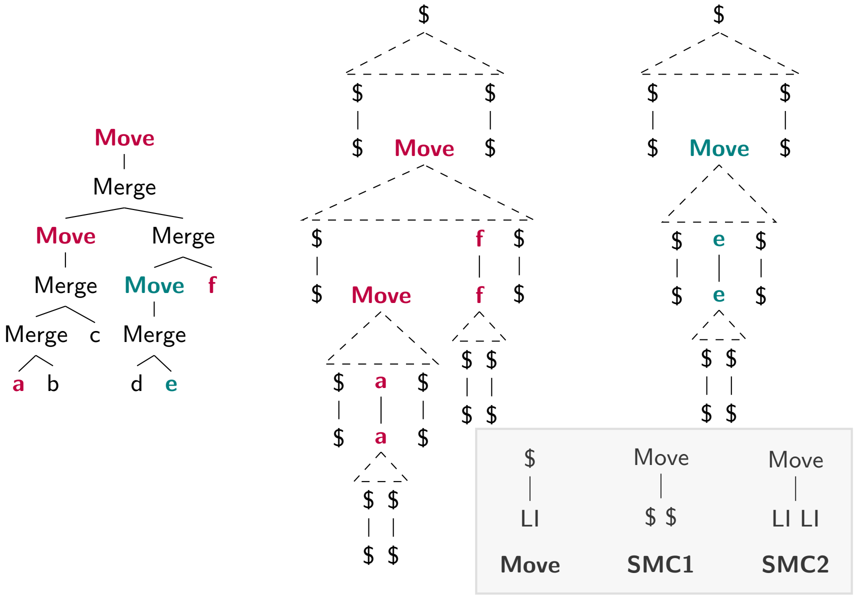




Example of Well-Formed Derivation



Example of Well-Formed Derivation



Conclusion

- The standard perspective views syntax and phonology as string-based algorithms of vastly different complexity.
- A linguistically informed perspective that picks more adequate data structures reveals profound similarities:
  - Phonology is tier-based strictly local over strings.
  - Syntax is tier-based strictly local over derivation trees.
- **Intuition**
  - Non-local dependencies are not particularly complex.
  - They are **local over a very simple relativization domain**.

References I

Chandlee, Jane. 2014. *Strictly local phonological processes*. Doctoral Dissertation, University of Delaware.

Doner, John. 1970. Tree acceptors and some of their applications. *Journal of Computer and System Sciences* 4:406–451.

Graf, Thomas. 2010. *Logics of phonological reasoning*. Master's thesis, University of California, Los Angeles.

Graf, Thomas. 2012. 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.

Heinz, Jeffrey. 2010. Learning long-distance phonotactics. *Linguistic Inquiry* 41:623–661.

Heinz, Jeffrey, Chetan Rawal, and Herbert G. Tanner. 2011. Tier-based strictly local constraints in phonology. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, 58–64.

Kaplan, Ronald M., and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics* 20:331–378.

References II

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.

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

Rogers, James. 2003. Syntactic structures as multi-dimensional trees. *Research on Language and Computation* 1:265–305.

Shieber, Stuart M. 1985. Evidence against the context-freeness of natural language. *Linguistics and Philosophy* 8:333–345.

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.

Thatcher, James W. 1967. Characterizing derivation trees for context-free grammars through a generalization of finite automata theory. *Journal of Computer and System Sciences* 1:317–322.