Tiers and Relativized Locality Across Language Modules

Thomas Graf
Aniello De Santo
Jon Rawski
Alëna Aksënova
Hossep Dolatian
Sedigheh Moradi
Hyunah Baek
Suji Yang
Jeffrey Heinz

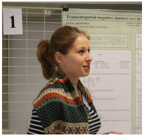
Stony Brook University
aniello.desanto@stonybrook.edu
https://aniellodesanto.github.io/

Parallels Between Phonology & Syntax
Amsterdam, July 9, 2018

The Subregular Group @ SBU

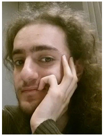
Jeff Heinz

Thomas Graf



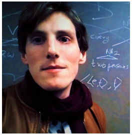
Alëna Aksënova

Hyunah Baek



Hossep Dolatian

Sedigheh Moradi



Jon Rawski

Suji Yang

The Elevator Pitch

Parallels between phonology and syntax?

- What would a computational linguist tell you?
  Well, it depends!
- What will I show you today?
  They are fundamentally similar!

The Take-Home Message

- **Two kind of dependencies**: local and non-local
- The core mechanisms are **the same** cross-domain, over the respective **structural representations**.
- **Relativized locality** plays a major role

The Elevator Pitch

Parallels between phonology and syntax?

- What would a computational linguist tell you?
  Well, it depends!
- What will I show you today?
  They are fundamentally similar!

The Take-Home Message

- **Two kind of dependencies**: local and non-local
- The core mechanisms are **the same** cross-domain, over the respective **structural representations**.
- **Relativized locality** plays a major role

The Elevator Pitch

Parallels between phonology and syntax?

- What would a computational linguist tell you?
  Well, it depends!

- What will I show you today?
  They are fundamentally similar!

The Take-Home Message

- **Two kind of dependencies**: local and non-local
- The core mechanisms are **the same** cross-domain, over the respective **structural representations**.
- **Relativized locality** plays a major role

Outline

A methodological note:

- Only phonotactics considered (no input-output mappings)
- Minimalist Grammars (Stabler 1997) as a model of syntax
- Formal language theory as a tool to assess parallelisms

Outline

1. **Local Dependencies**
   - In Phonology
   - In Syntax
2. **Non-local Dependencies**
   - In Phonology
   - In Syntax

A methodological note:

- Only phonotactics considered (no input-output mappings)
- Minimalist Grammars (Stabler 1997) as a model of syntax
- Formal language theory as a tool to assess parallelisms

Local Dependencies in Phonology

1. **Word-final devoicing**
   Forbid voiced segments at the end of a word

   (1) a. * rad
   b. rat

1. **Intervocalic voicing**
   Forbid voiceless segments in between two vowels

   (2) a. * faser
   b. fazer

These patters can be described by **strictly local** (SL) constraints.

Local Dependencies in Phonology

1. **Word-final devoicing**
   Forbid voiced segments at the end of a word

(1) a. * rad
    b. rat

1. **Intervocalic voicing**
   Forbid voiceless segments in between two vowels

(2) a. * faser
    b. fazer

These patters can be described by **strictly local** (SL) constraints.

Local Dependencies in Phonology are SL

Example: Word-final devoicing

- Forbid voiced segments at the end of a word: *[+voice]$
- **German**: ***z**$, ***v**$,***d**$ ($ = word edge).

$ r a **d** $ $ r a t $

Example: Intervocalic voicing

- Forbid voicess segments in-between two vowels: *V[-voice]V
- **German**: *a**s**e, *i**s**e, *e**s**e, *i**s**i, . . .

$ f a **s** e r $ $ f a z e r $

Local Dependencies in Phonology are SL

Example: Word-final devoicing

- Forbid voiced segments at the end of a word: *[+voice]$
- **German**: ***z**$, ***v**$,***d**$ ($ = word edge).

* $ r a **d** $ *ok* $ r a t $

Example: Intervocalic voicing

- Forbid voicess segments in-between two vowels: *V[-voice]V
- **German**: *a**s**e, *i**s**e, *e**s**e, *i**s**i, . . .

$ f a **s** e r $ $ f a z e r $

Local Dependencies in Phonology are SL

Example: Word-final devoicing

- Forbid voiced segments at the end of a word: *[+voice]$
- **German**: *z$, *v$,*d$ ($ = word edge).

* $ r a [d $] *ok* $ r a [t $]

Example: Intervocalic voicing

- Forbid voicess segments in-between two vowels: *V[-voice]V
- **German**: *ase, *ise, *ese, *isi, . . .

* $ f [a s e] r $ *ok* $ f [a z e] r $

What about Syntax?

We need a model for syntax ...

- Minimalist grammars (MGs) are a formalization of Minimalist syntax. (Stabler 1997, 2011)
- Operations: **Merge** and **Move**
- Adopt Chomsky-Borer hypothesis:
  Grammar is just a finite list of feature-annotated lexical items

Local dependencies in syntax

- Merge is a **feature-driven** operation:
  category feature N^-, D^-, ...
  selector feature N^+, D^+, ...
- Subcategorization as formalized by Merge is **strictly local**.

What about Syntax?

We need a model for syntax ...

- Minimalist grammars (MGs) are a formalization of Minimalist syntax. (Stabler 1997, 2011)
- Operations: **Merge** and **Move**
- Adopt Chomsky-Borer hypothesis:
  Grammar is just a finite list of feature-annotated lexical items

Local dependencies in syntax

- Merge is a **feature-driven** operation:
  category feature N^{-}, D^{-}, ...
  selector feature N^{+}, D^{+}, ...
- Subcategorization as formalized by Merge is **strictly local**.

Local Dependencies in Syntax

Merge is a **feature-driven** operation:

- category feature N^-, D^-, ...
- selector feature N^+, D^+, ...

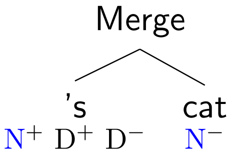
's $\mathrm{N}^+ \ \mathrm{D}^+ \ \mathrm{D}^-$

cat N^-

Local Dependencies in Syntax

Merge is a **feature-driven** operation:

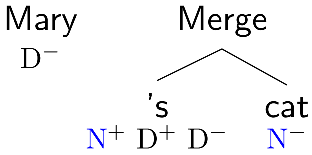
- category feature N^-, D^-, ...
- selector feature N^+, D^+, ...



Local Dependencies in Syntax

Merge is a **feature-driven** operation:

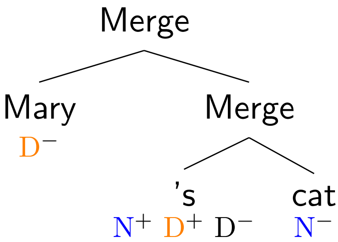
- category feature N^-, D^-, ...
- selector feature N^+, D^+, ...



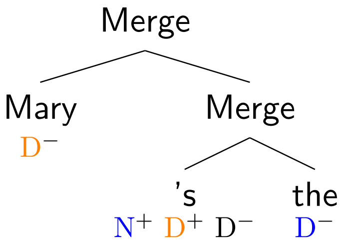
Local Dependencies in Syntax

Merge is a **feature-driven** operation:

- category feature N^-, D^-, ...
- selector feature N^+, D^+, ...

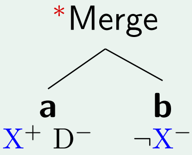


Merge is SL (Graf 2012a)

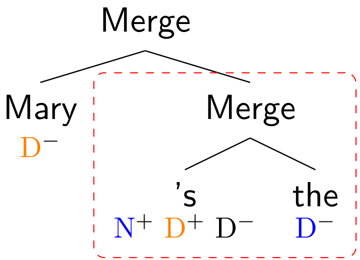


SL constraints on Merge

- We lift constraints from string n-grams to tree n-grams
- We get SL constraints over subtrees.

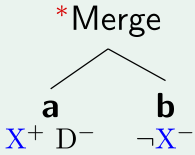


Merge is SL (Graf 2012a)



SL constraints on Merge

- We lift constraints from string n-grams to tree n-grams
- We get SL constraints over subtrees.



Interim Summary

	Local	Data Structure
Phonology	?	?
Syntax	?	?

Local phenomena modeled by n-grams of bounded size:

- computationally very simple
- learnable from positive examples of strings/trees
- plausible cognitive requirements

Interim Summary

	Local	Data Structure
Phonology	**SL**	**Strings**
Syntax	**SL**	**Trees**

Local phenomena modeled by n-grams of bounded size:

- computationally very simple
- learnable from positive examples of strings/trees
- plausible cognitive requirements

Interim Summary

	Local	Non-local	Data Structure
Phonology	**SL**	?	**Strings**
Syntax	**SL**	?	**Trees**

Local phenomena modeled by n-grams of bounded size:

- computationally very simple
- learnable from positive examples of strings/trees
- plausible cognitive requirements

Unbounded Dependencies in Phonology

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

(3) a. * hasxintilawaʃ
  b. * haʃxintilawas
  c. haʃxintilawaʃ

- **Unbounded Tone Plateauing in Luganda** (UTP)
  No L may occur within an interval spanned by H.
  (Hyman 2011)

(4) a. **LHLLLL**
  b. **LLLLHL**
  c. * **LHLLHL**
  d. **LHHHHL**

Unbounded Dependencies Are Not SL

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

(5) a. * ha**s**xintilawa**ʃ**
b. * ha**ʃ**xintilawa**s**
c. ha**ʃ**xintilawa**ʃ**

Example: Samala

* $ h a **s** x i n t i l a w a **ʃ** $

$ h a **ʃ** x i n t i l a w a **ʃ** $

Unbounded Dependencies Are Not SL

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

(5)
  a. * hasxintilawaʃ
  b. * haʃxintilawas
  c. haʃxintilawaʃ

Example: Samala

*$ h a s x i n t i l a w a ʃ $

$ h a ʃ x i n t i l a w a ʃ $

Unbounded Dependencies Are Not SL

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

(5) a. * hasxintilawaʃ
b. * haʃxintilawas
c. haʃxintilawaʃ

Example: Samala

* $ h a s x i n t i l a w a ʃ $

$ h a ʃ x i n t i l a w a ʃ $

Unbounded Dependencies Are Not SL

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

(5) a. * hasxintilawaʃ
b. * haʃxintilawas
c. haʃxintilawaʃ

Example: Samala

* $ h a [s x i n t i l a w a ʃ] $

$ h a [ʃ x i n t i l a w a ʃ] $

Unbounded Dependencies Are Not SL

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

(5) a. * hasxintilawaʃ
b. * haʃxintilawas
c. haʃxintilawaʃ

Example: Samala

* $ h a s x i n t i l a w a ʃ $

$ h a ʃ x i n t i l a w a ʃ $

- **But:** Sibilants can be arbitrarily far away from each other!

* $ s t a j a n o w o n w a ʃ $

Unbounded Dependencies Are Not SL

- **Samala Sibilant Harmony**
  Sibilants must not disagree in anteriority.
  (Applegate 1972)

(5) a. * hasxintilawaʃ
 b. * haʃxintilawas
 c. haʃxintilawaʃ

Example: Samala

* $ h a s x i n t i l a w a ʃ $

$ h a ʃ x i n t i l a w a ʃ $

- **But:** Sibilants can be arbitrarily far away from each other!

* $ s t a j a n o w o n w a ʃ $

Locality Over Tiers

* $ **s** t a j a n o w o n w a **ʃ** $

- Sibilants can be arbitrarily far away from each other!
- **Problem**: SL limited to locality domains of size n;

Tier-based Strictly Local (TSL) Grammars (Heinz et al. 2011)

- Projection of selected segments on a tier T;
- Strictly local constraints over T determine wellformedness;
- Unbounded dependencies are local over **tiers**.

Locality Over Tiers

* $ s t a j a n o w o n w a ʃ $

- Sibilants can be arbitrarily far away from each other!
- **Problem**: SL limited to locality domains of size n;

Tier-based Strictly Local (TSL) Grammars (Heinz et al. 2011)

- Projection of selected segments on a tier T;
- Strictly local constraints over T determine wellformedness;
- Unbounded dependencies are local over **tiers**.

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * ha**s**xintilawa**ʃ**

b. * ha**ʃ**xintilawa**s**

c. ha**ʃ**xintilawa**ʃ**

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

* $ h a **s** x i n t i l a w **ʃ** $ ^{ok} $ h a **ʃ** x i n t i l a w **ʃ** $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ
    b. * haʃxintilawas
    c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

* $ h a s x i n t i l a w ʃ $ ^{ok} $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ
b. * haʃxintilawas
c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

* $ h a s x i n t i l a w ʃ $

ok $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ

b. * haʃxintilawas

c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

* $ h a s x i n t i l a w ʃ $ ok $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ
b. * haʃxintilawas
c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

s

* $ h a s x i n t i l a w ʃ $ 		ok $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ
b. * haʃxintilawas
c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

s

* $ h a s x i n t i l a w ʃ $ ^{ok} $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ
    b. * haʃxintilawas
    c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

s

* $ h a s x i n t i l a w ʃ $ ok $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * ha**s**xintilawa**ʃ**
 b. * ha**ʃ**xintilawa**s**
 c. ha**ʃ**xintilawa**ʃ**

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

s

* $ h a s x i n t i l a w ʃ $ ^{ok} $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6)
a. * ha**s**xintilawa**ʃ**
b. * ha**ʃ**xintilawa**s**
c. ha**ʃ**xintilawa**ʃ**

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

s

* $ h a **s** x i n t i l a w **ʃ** $

ok $ h a **ʃ** x i n t i l a w **ʃ** $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ
b. * haʃxintilawas
c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

s

* $ h a s x i n t i l a w ʃ $ $\quad$ ^{ok} $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ
b. * haʃxintilawas
c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

s

* $ h a s x i n t i l a w ʃ $ ^{ok} $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ

b. * haʃxintilawas

c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

s

* $ h a s x i n t i l a w ʃ $ ok $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ

b. * haʃxintilawas

c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

s

* $ h a s x i n t i l a w ʃ $ $\quad$ ^{ok} $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6)
a. * hasxintilawaʃ
b. * haʃxintilawas
c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

s

* $ h a s x i n t i l a w ʃ $ ᵒᵏ $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6)
a. * ha**s**xintilawa**ʃ**
b. * ha**ʃ**xintilawa**s**
c. ha**ʃ**xintilawa**ʃ**

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]

Example: TSL Samala

s ʃ

* $ h a s x i n t i l a w ʃ $
ok $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ
    b. * haʃxintilawas
    c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]
  I.E. *sʃ, *sʒ, *zʃ, *zʒ, *ʃs, *ʒs, *ʃz, *ʒz

Example: TSL Samala

s ʃ

* $ h a s x i n t i l a w ʃ $ ok $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ
b. * haʃxintilawas
c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]
I.E. *sʃ, *sʒ, *zʃ, *zʒ, *ʃs, *ʒs, *ʃz, *ʒz

Example: TSL Samala

s ʃ

* $ h a s x i n t i l a w ʃ $

ok $ h a ʃ x i n t i l a w ʃ $

Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ
b. * haʃxintilawas
c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]
I.E. *sʃ, *sʒ, *zʃ, *zʒ, *ʃs, *ʒs, *ʃz, *ʒz

Example: TSL Samala

s ʃ

* $ h a s x i n t i l a w ʃ $

ʃ ʃ

ok $ h a ʃ x i n t i l a w ʃ $

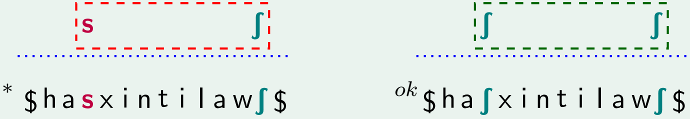
Unbounded Dependencies are TSL

- Let's revisit Samala Sibilant Harmony

(6) a. * hasxintilawaʃ
b. * haʃxintilawas
c. haʃxintilawaʃ

- What do we need to project? [+strident]
- What do we need to ban? *[+ant][−ant],*[−ant][+ant]
  I.E. *sʃ, *sʒ, *zʃ, *zʒ, *ʃs, *ʒs, *ʃz, *ʒz

Example: TSL Samala



TSL Phonology: Accounting for Context

- **Unbounded Tone Plateauing in Luganda** (UTP)
  No L may occur within an interval spanned by H.
  (Hyman 2011)

(7)
a. LHLLLL
b. LLLLHL
c. *LHLLHL
d. LHHHHL

Example

TSL Phonology: Accounting for Context

- **Unbounded Tone Plateauing in Luganda** (UTP)
  No L may occur within an interval spanned by H.
  (Hyman 2011)

(7) a. LHLLLL
b. LLLLHL
c. *LHLLHL
d. LHHHHL

Example

TSL Phonology: Accounting for Context

- **Unbounded Tone Plateauing in Luganda** (UTP)
  No L may occur within an interval spanned by H.
  (Hyman 2011)

(7)
a. LHLLLL
b. LLLLHL
c. *LHLLHL
d. LHHHHL

Example

*L H L L H L

TSL Phonology: Accounting for Context

- **Unbounded Tone Plateauing in Luganda** (UTP)
  No L may occur within an interval spanned by H.
  (Hyman 2011)

(7)
a. LHLLLL
b. LLLLHL
c. *LHLLHL
d. LHHHHL

Example

L H L L H L

....

*L H L L H L

TSL Phonology: Accounting for Context

- **Unbounded Tone Plateauing in Luganda** (UTP)
  No L may occur within an interval spanned by H.
  (Hyman 2011)

(7)
a. LHLLLL
b. LLLLHL
c. *LHLLHL
d. LHHHHL

Example

L H L L H L

*L H L L H L

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

^{ok}**L H L L L L** $\quad$ ***L H L L H L**

- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

H

....

^{ok} **L H L L L L** $\quad$ * **L H L L H L**

- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

H L

^{ok}**L H L L L L**

****L H L L H L**

- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

H L

^{ok}**L H L L L L** $\quad$ ***L H L L H L**

- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

H L

^{ok}**L H L L L L** *L H L L H L

- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

H L

....

^{ok}**L H L L L L** ***L H L L H L**

- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

HL

^{ok}**L H L L L L** ***L H L L H L**

- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

H L

^{ok}**L H L L L L**

***L H L L H L**

- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

H L	H
ok L H L L L L	* L H L L H L

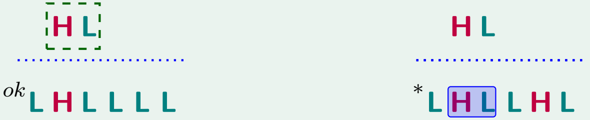
- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example



- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

H L	**H L**
^{ok}**L H L L L L**	***L H L L H L**

- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

H L	**H L H**
^{ok}**L H L L L L**	***L H L L H L**

- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

H L	**H L H L**
^{ok}**L H L L L L**	***L H L L H L**

- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example

H L

^{ok} L H L L L L

H L H L

* L H L L H L

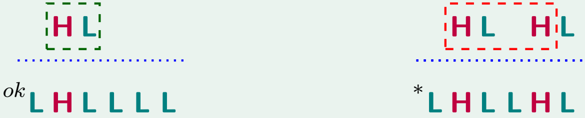
- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Accounting for Context [cont.]

A TSL analysis for UTP (De Santo and Graf 2017):

- Project every **H**; project **L** iff immediately follows **H**
- Ban: **HLH**

Example



- Most non-local dependencies in phonology are TSL
- **What about syntax?**

Non-Local Dependencies in Syntax

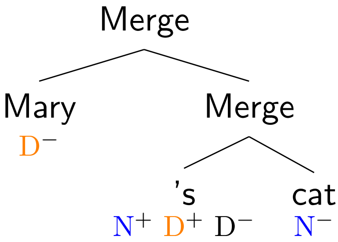
Let's stick to core operations:

- Move
- Merge?

Non-Local Dependencies in Syntax

Let's stick to core operations:

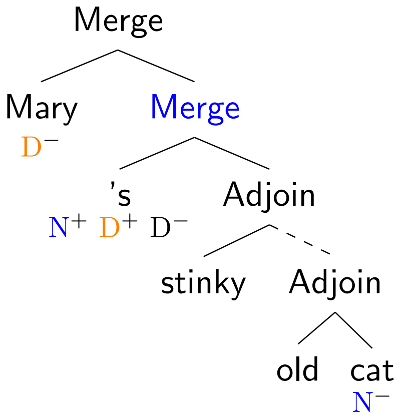
- Move
- Merge?



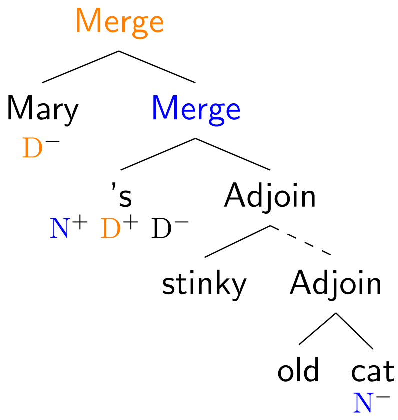
Non-Local Dependencies in Syntax

Let's stick to core operations:

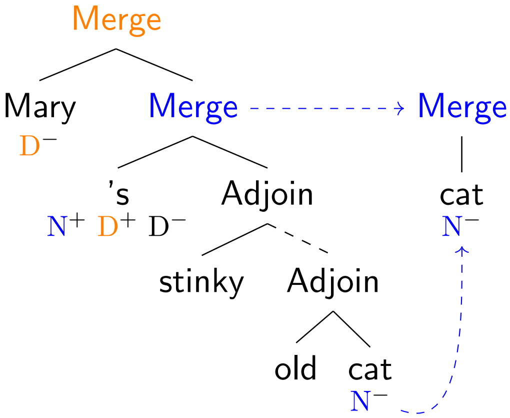
- Move
- **Merge**: Unbounded adjunction
  Frey and Gärtner (2002); Graf (2017b)



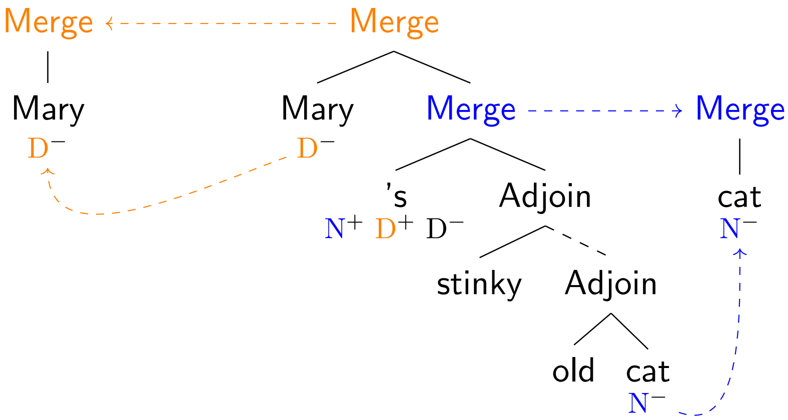
TSL over Trees: Projecting Tiers



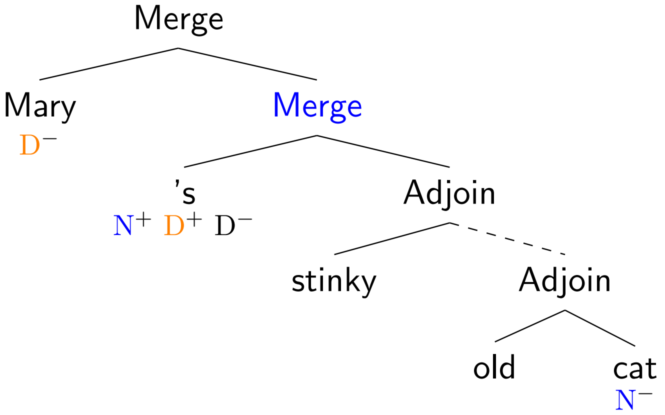
TSL over Trees: Projecting Tiers



TSL over Trees: Projecting Tiers

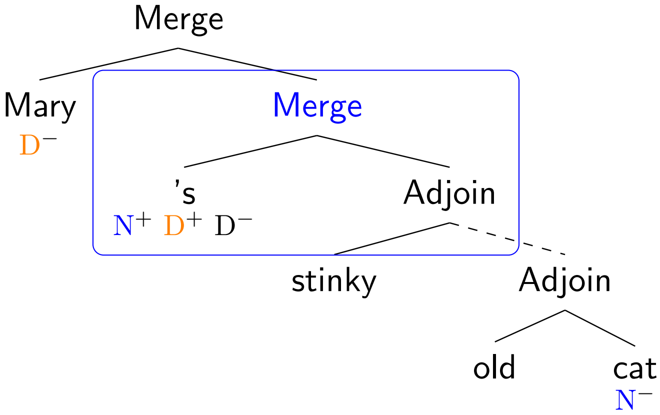


Merge with Adjunction is TSL



A TSL grammar for Merge

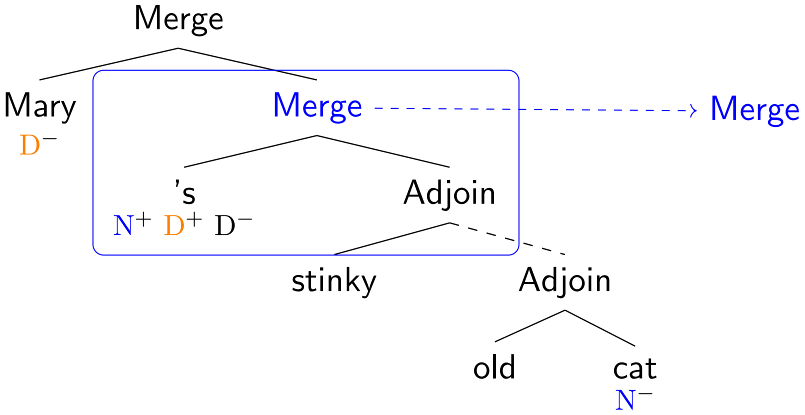
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^+ (e.g. $\mathrm{X} = \mathrm{N}$)

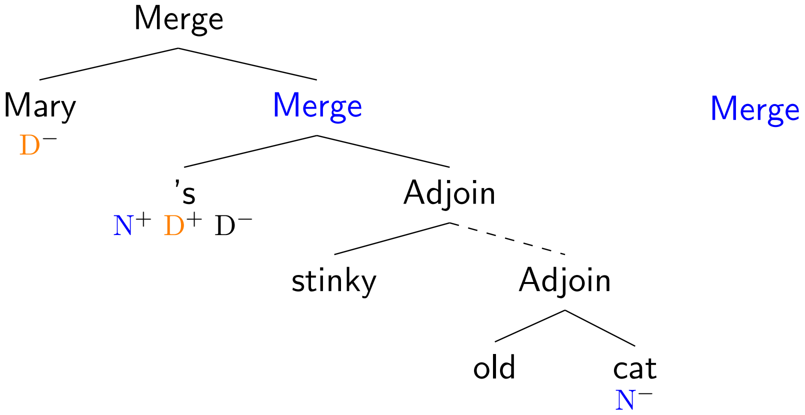
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^+ (e.g. $\mathrm{X} = \mathrm{N}$)

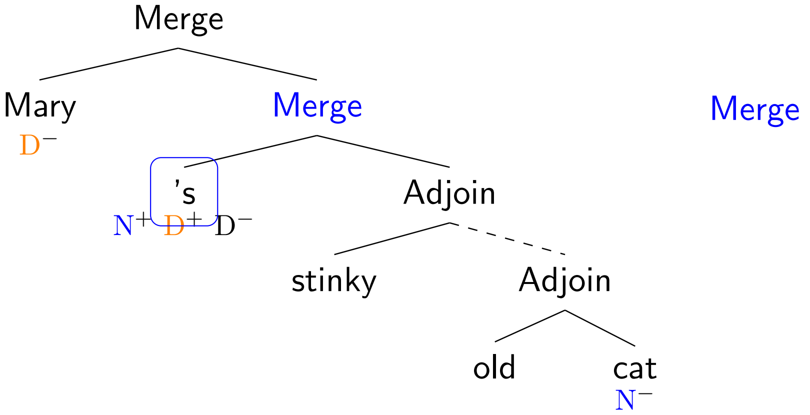
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^{+} (e.g. $\mathrm{X} = \mathrm{N}$)
2. Project any node which has X^{-} (e.g. $\mathrm{X} = \mathrm{N}$)

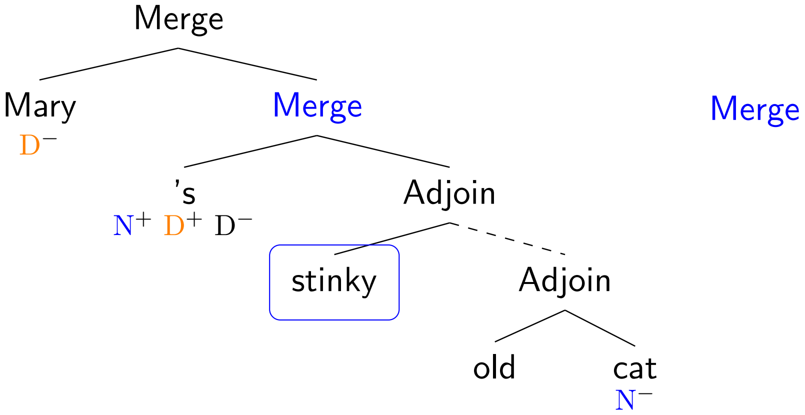
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^+ (e.g. $X = N$)
2. Project any node which has X^- (e.g. $X = N$)

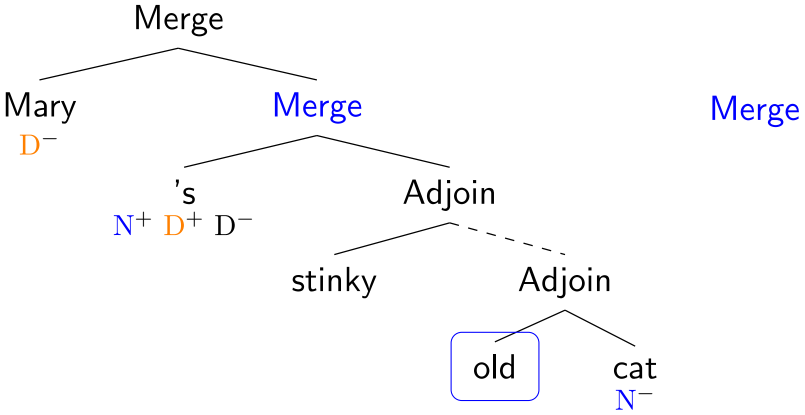
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^+ (e.g. $\mathrm{X} = \mathrm{N}$)
2. Project any node which has X^- (e.g. $\mathrm{X} = \mathrm{N}$)

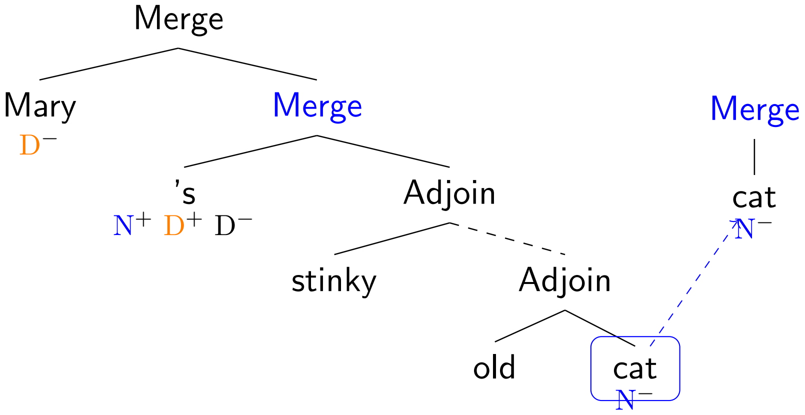
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^+ (e.g. $X = N$)
2. Project any node which has X^- (e.g. $X = N$)

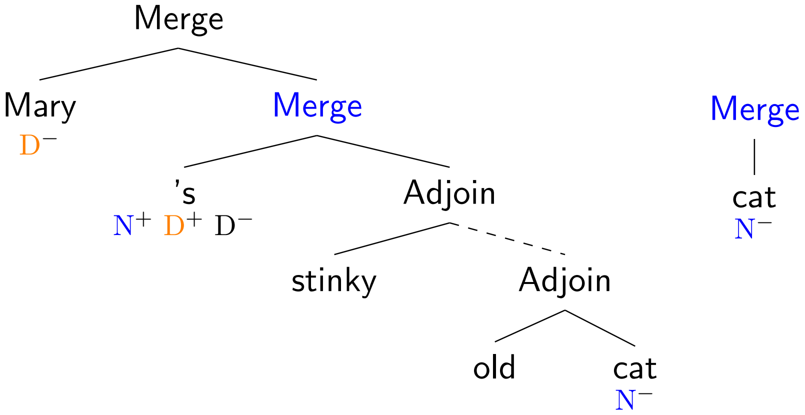
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^{+} (e.g. $\mathrm{X} = \mathrm{N}$)
2. Project any node which has X^{-} (e.g. $\mathrm{X} = \mathrm{N}$)

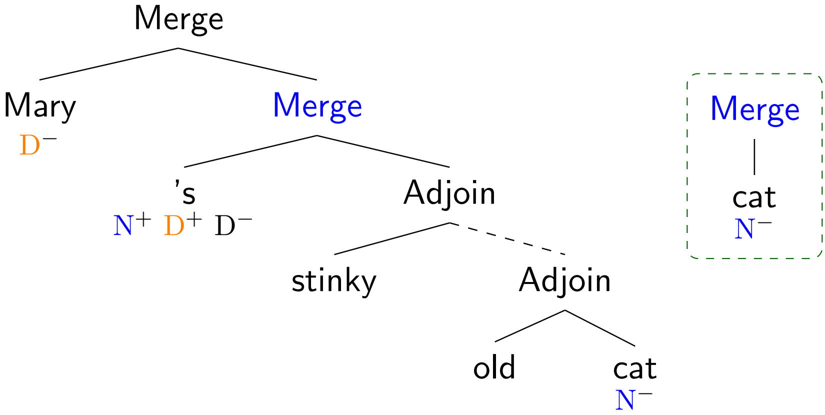
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^+ (e.g. $X = N$)
2. Project any node which has X^- (e.g. $X = N$)

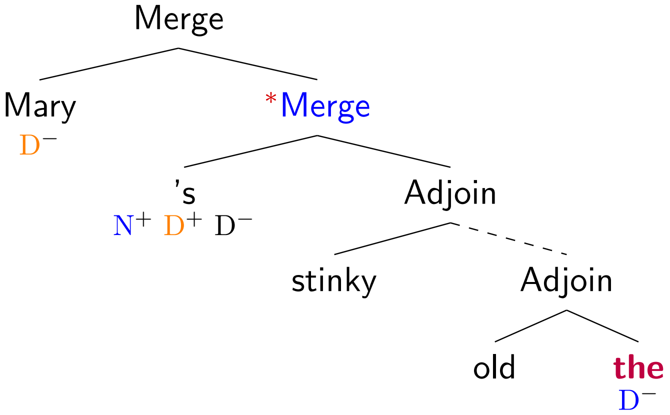
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^+ (e.g. $X = N$)
2. Project any node which has X^- (e.g. $X = N$)
3. No Merge without exactly one LI among its daughters.

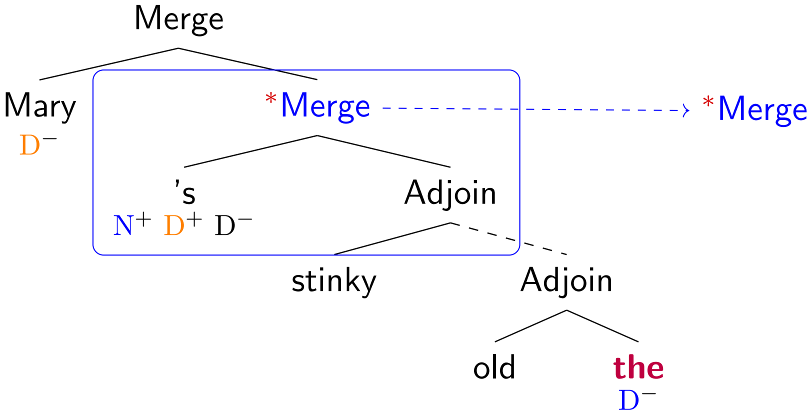
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^+ (e.g. $\mathrm{X} = \mathrm{V}$)
2. Project any node which has X^- (e.g. $\mathrm{X} = \mathrm{V}$)
3. No Merge without exactly one LI among its daughters.

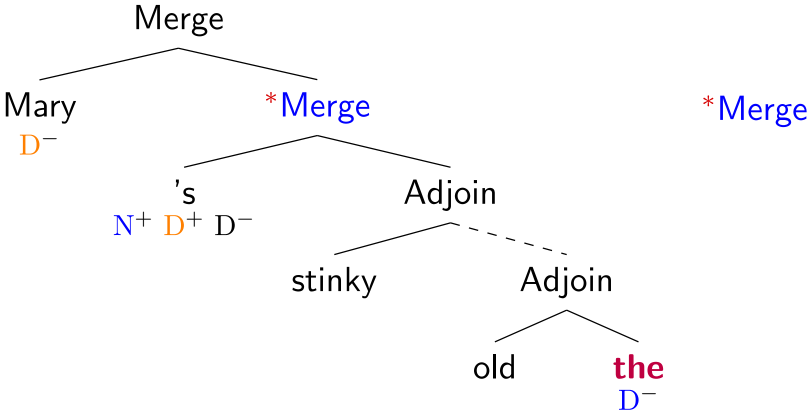
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^{+} (e.g. $\mathrm{X} = \mathrm{V}$)
2. Project any node which has X^{-} (e.g. $\mathrm{X} = \mathrm{V}$)
3. No Merge without exactly one LI among its daughters.

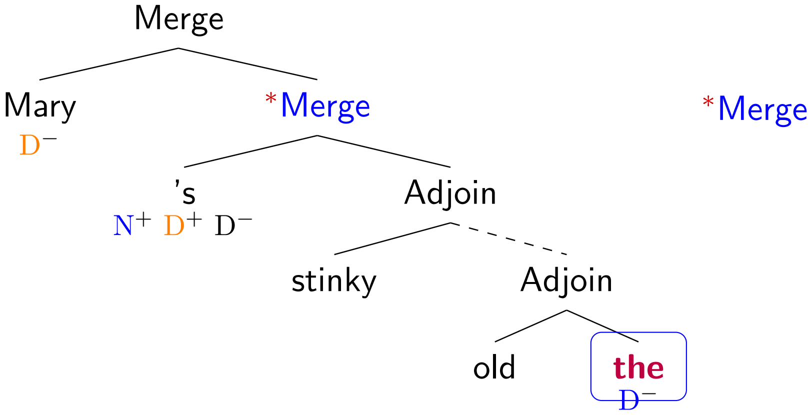
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^- (e.g. $X = V$)
2. Project any node which has X^+ (e.g. $X = V$)
3. No Merge without exactly one LI among its daughters.

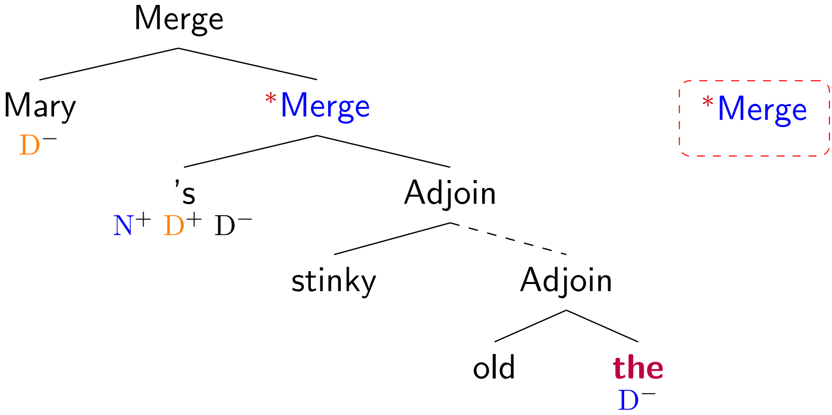
Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^- (e.g. $\mathrm{X} = \mathrm{V}$)
2. Project any node which has X^+ (e.g. $\mathrm{X} = \mathrm{V}$)
3. No Merge without exactly one LI among its daughters.

Merge with Adjunction is TSL



A TSL grammar for Merge

1. Project **Merge** iff a child has X^- (e.g. $X = V$)
2. Project any node which has X^+ (e.g. $X = V$)
3. No Merge without exactly one LI among its daughters.

Parallels Between Phonology And Syntax

	Local	**Non-local**
Phonology	?	?
Syntax	?	?

- **Relativized Locality:**
  Non-local dependencies are local over a simple relativization domain.

Strong Cognitive Parallelism Hypothesis

Phonology, (morphology), and syntax have the **same subregular complexity** over their respective **structural representations**.

Parallels Between Phonology And Syntax

	Local	Non-local
Phonology	**SL**	?
Syntax	**SL**	?

- **Relativized Locality:**
  Non-local dependencies are local over a simple relativization domain.

Strong Cognitive Parallelism Hypothesis

Phonology, (morphology), and syntax have the **same subregular complexity** over their respective **structural representations**.

Parallels Between Phonology And Syntax

	Local	Non-local
Phonology	**SL**	**TSL**
Syntax	**SL**	**TSL**

- **Relativized Locality:**
  Non-local dependencies are local over a simple relativization domain.

Strong Cognitive Parallelism Hypothesis

Phonology, (morphology), and syntax have the **same subregular complexity** over their respective **structural representations**.

Parallels Between Phonology And Syntax

	Local	Non-local	Data Structure
Phonology	**SL**	**TSL**	**Strings**
Syntax	**SL**	**TSL**	**Trees**

- **Relativized Locality:**
  Non-local dependencies are local over a simple relativization domain.

Strong Cognitive Parallelism Hypothesis

Phonology, (morphology), and syntax have the **same subregular complexity** over their respective **structural representations**.

Parallels Between Phonology And Syntax

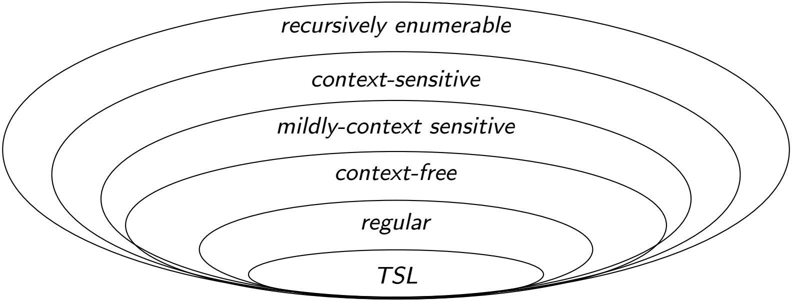
	Local	Non-local	Data Structure
Phonology	SL	TSL	Strings
Syntax	SL	TSL	Trees

- **Relativized Locality:**
  Non-local dependencies are local over a simple relativization domain.

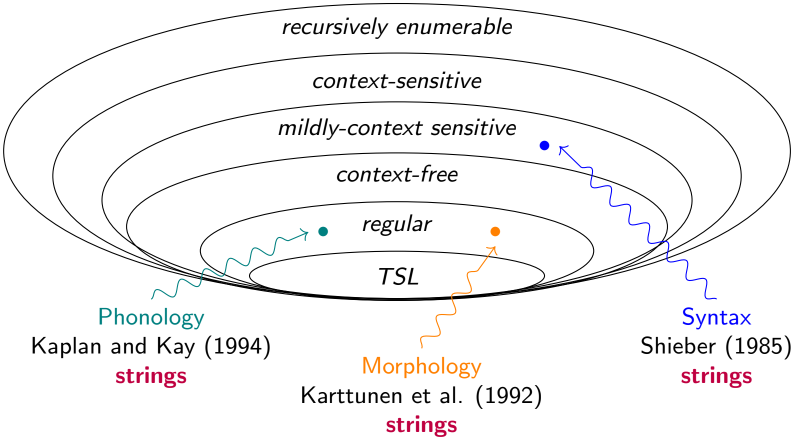
Strong Cognitive Parallelism Hypothesis

Phonology, (morphology), and syntax have the **same subregular complexity** over their respective **structural representations**.

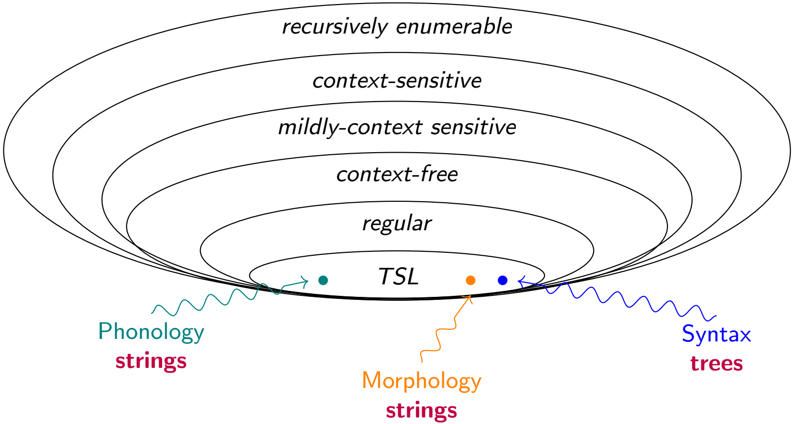
A Bird's-Eye View of the Framework



A Bird's-Eye View of the Framework



A Bird's-Eye View of the Framework



Conclusion

Strong Cognitive Parallelism Hypothesis

Phonology, (morphology), and syntax have the **same subregular complexity** over their respective **structural representations**.

We gain a unified perspective on:

- typology
- learnability
- cognition

Conclusion

Strong Cognitive Parallelism Hypothesis

Phonology, (morphology), and syntax have the **same subregular complexity** over their respective **structural representations**.

We gain a unified perspective on:

- typology
  - × Intervocalic Voicing iff applied an even times in the string
  - × Have a CP iff it dominates ≥ 3 TPs
- learnability
- cognition

Conclusion

Strong Cognitive Parallelism Hypothesis

Phonology, (morphology), and syntax have the **same subregular complexity** over their respective **structural representations**.

We gain a unified perspective on:

- typology
  - × Intervocalic Voicing iff applied an even times in the string
  - × Have a CP iff it dominates ≥ 3 TPs
- learnability

  Learnable from positive examples of strings/trees.
- cognition

Conclusion

Strong Cognitive Parallelism Hypothesis

Phonology, (morphology), and syntax have the **same subregular complexity** over their respective **structural representations**.

We gain a unified perspective on:

- typology
  - × Intervocalic Voicing iff applied an even times in the string
  - × Have a CP iff it dominates ≥ 3 TPs
- learnability

  Learnable from positive examples of strings/trees.
- cognition

  Finite, flat memory

Future Work

We are just getting started:

- autosegmental structures (Jardine 2017:i.a)
- morphological derivations (Chandlee 2017; Aksënova and De Santo 2017)
- mappings (Chandlee 2014; Chandlee and Heinz 2018:i.a.)
- syntax beyond Merge and Move (Graf 2017b; Vu 2018)

Join the Enterprise!

- typological universals/gaps
- TSL-analyses of phenomena/counterexamples
- artificial language learning experiments
- new formal results
- and much more ...

References I

Aksënova, Alëna, and Aniello De Santo. 2017. Strict locality in morphological derivations. In *Proceedings of the 53rd Meeting of the Chicago Linguistic Society (CLS53)*. (to appear).

Aksënova, Alëna, Thomas Graf, and Sedigheh Moradi. 2016. Morphotactics as tier-based strictly local dependencies. In *Proceedings of SIGMorPhon 2016*. To appear.

Applegate, R.B. 1972. *Ineseno chumash grammar*. Doctoral Dissertation, University of California,Berkeley.

Chandlee, Jane. 2014. *Strictly local phonological processes*. Doctoral Dissertation, University of Delaware. URL http://udspace.udel.edu/handle/19716/13374.

Chandlee, Jane. 2017. Computational locality in morphological maps. *Morphology* 27:599–641.

Chandlee, Jane, and Jeffrey Heinz. 2018. Strict locality and phonological maps. *Linguistic Inquiry* 49:23–60.

De Santo, Aniello, and Thomas Graf. 2017. Structure sensitive tier projection: Applications and formal properties. Ms., Stony Brook University.

Epstein, Samuel D., Erich M. Groat, Ruriko Kawashima, and Hisatsugu Kitahara. 1998. *A derivational approach to syntactic relations*. Oxford: Oxford University Press.

References II

Fowlie, Meaghan. 2013. Order and optionality: Minimalist grammars with adjunction. In *Proceedings of the 13th Meeting on the Mathematics of Language (MoL 13)*, ed. András Kornai and Marco Kuhlmann, 12–20.

Frey, Werner, and Hans-martin Gärtner. 2002. On the treatment of scrambling and adjunction in minimalist grammars. In *In Proceedings, Formal Grammar?02*. Citeseer.

Gärtner, Hans-Martin, and Jens Michaelis. 2010. On the treatment of multiple-wh-interrogatives in Minimalist grammars. In *Language and logos*, ed. Thomas Hanneforth and Gisbert Fanselow, 339–366. Berlin: Akademie Verlag.

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. URL `http://dx.doi.org/10.1007/978-3-642-32024-8_14`.

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. URL `http://dx.doi.org/10.1007/978-3-642-31262-5_4`.

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.

References III

Graf, Thomas. 2013. *Local and transderivational constraints in syntax and semantics*. Doctoral Dissertation, UCLA. URL http://thomasgraf.net/doc/papers/PhDThesis_RollingRelease.pdf.

Graf, Thomas. 2014a. Late merge as lowering movement in Minimalist grammars. In *LACL 2014*, ed. Nicholas Asher and Sergei Soloviev, volume 8535 of *Lecture Notes in Computer Science*, 107–121. Heidelberg: Springer.

Graf, Thomas. 2014b. Models of adjunction in Minimalist grammars. In *Formal Grammar 2014*, ed. Glynn Morrill, Reinhard Muskens, Rainer Osswald, and Frank Richter, volume 8612 of *Lecture Notes in Computer Science*, 52–68. Heidelberg: Springer.

Graf, Thomas. 2017a. Grammar size and quantitative restrictions on movement. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2018*, 23–33.

Graf, Thomas. 2017b. Why movement comes for free once you have adjunction. In *Proceedings of CLS 53*. URL http://ling.auf.net/lingbuzz/003943, (to appear).

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. URL http://www.aclweb.org/anthology/P11-2011.

Hunter, Tim. 2015. Deconstructing merge and move to make room for adjunction. *Syntax* 18:266–319.

References IV

Hunter, Tim, and Robert Frank. 2014. Eliminating rightward movement: Extraposition as flexible linearization of adjuncts. *Linguistic Inquiry* 45:227–267.

Hyman, Larry M. 2011. Tone: Is it different? *The Handbook of Phonological Theory, Second Edition* 197–239.

Jardine, Adam. 2017. On the logical complexity of autosegmental representations. In *Proceedings of the 15th Meeting on the Mathematics of Language*, ed. Makoto Kanazawa, Philippe de Groote, and Mehrnoosh Sadrzadeh, 22–35. London, UK: Association for Computational Linguistics.

Kobele, Gregory M. 2006. *Generating copies: An investigation into structural identity in language and grammar*. Doctoral Dissertation, UCLA. URL `http://home.uchicago.edu/~gkobele/files/Kobele06GeneratingCopies.pdf`.

Kobele, Gregory M. 2008. Across-the-board extraction and Minimalist grammars. In *Proceedings of the Ninth International Workshop on Tree Adjoining Grammars and Related Frameworks*.

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

Kobele, Gregory M. 2011. Minimalist tree languages are closed under intersection with recognizable tree languages. In *LACL 2011*, ed. Sylvain Pogodalla and Jean-Philippe Prost, volume 6736 of *Lecture Notes in Artificial Intelligence*, 129–144.

References V

Kobele, Gregory M., Sabrina Gerth, and John T. Hale. 2012. Memory resource allocation in top-down Minimalist parsing. In *Proceedings of Formal Grammar 2012*.

McMullin, Kevin. 2016. *Tier-based locality in long-distance phonotactics: Learnability and typology*. Doctoral Dissertation, Uniersity of British Columbia.

Pasternak, Robert. 2016. Memory usage and scope ambiguity resolution. Qualifying paper, Stony Brook University.

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.

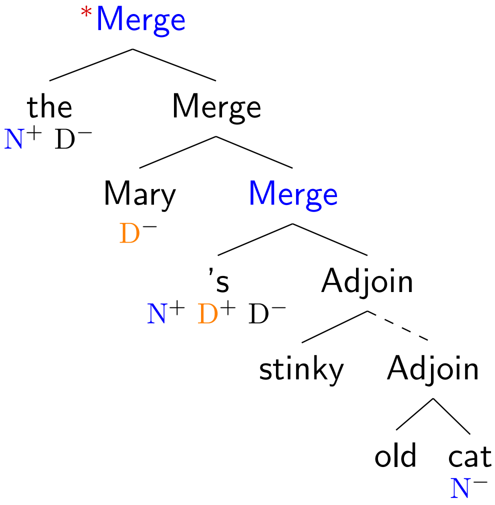
Stabler, Edward P. 2006. Sidewards without copying. In *Formal Grammar '06, Proceedings of the Conference*, ed. Gerald Penn, Giorgio Satta, and Shuly Wintner, 133–146. Stanford: CSLI.

Stabler, Edward P. 2011. Computational perspectives on Minimalism. In *Oxford handbook of linguistic Minimalism*, ed. Cedric Boeckx, 617–643. Oxford: Oxford University Press.

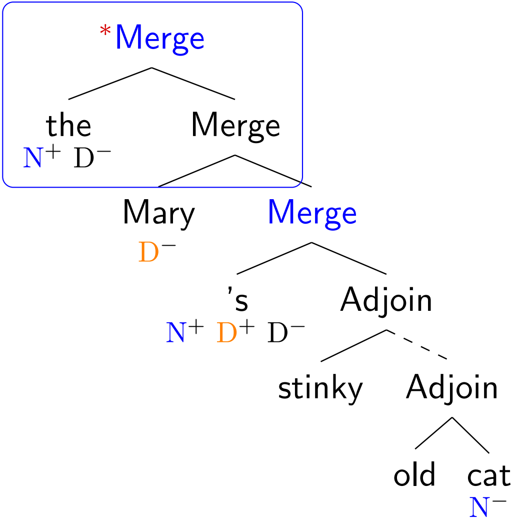
Stabler, Edward P. 2013. Two models of minimalist, incremental syntactic analysis. *Topics in Cognitive Science* 5:611–633.

Vu, Mai Ha. 2018. Towards a formal description of npi-licensing patterns. *Proceedings of the Society for Computation in Linguistics (SCiL) 2018* 154–163.

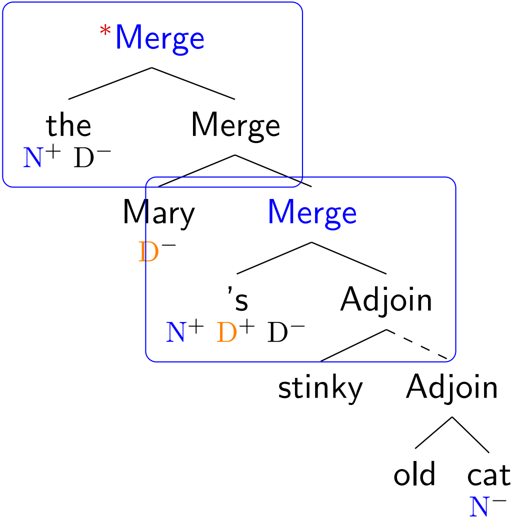
TSL Merge: Understanding the Constraint



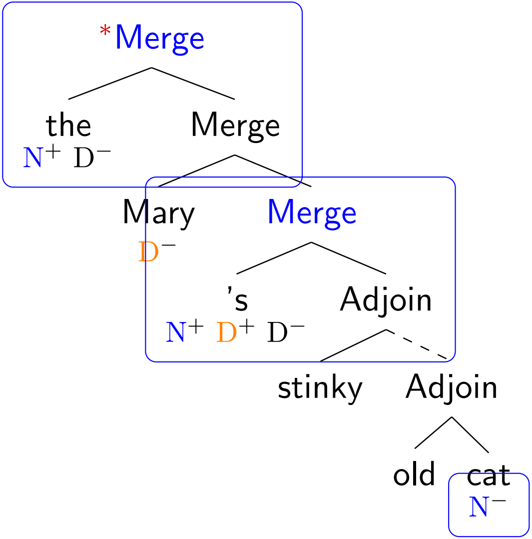
TSL Merge: Understanding the Constraint



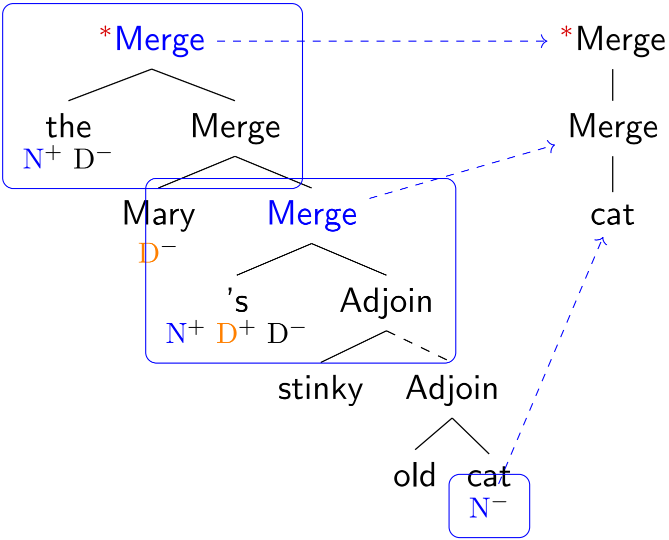
TSL Merge: Understanding the Constraint



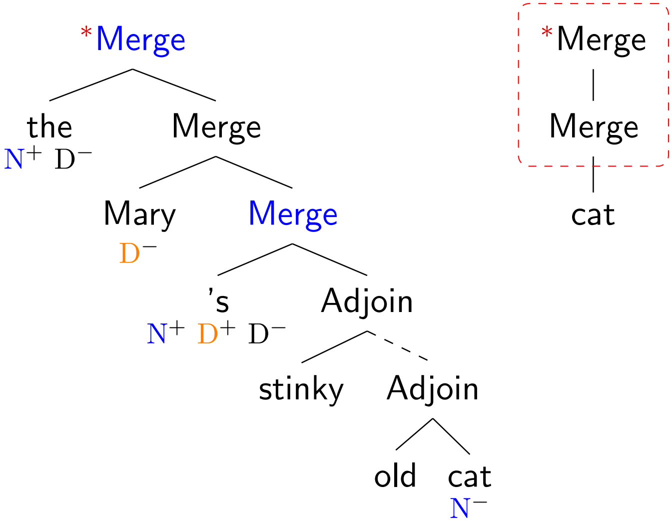
TSL Merge: Understanding the Constraint



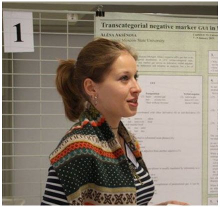
TSL Merge: Understanding the Constraint



TSL Merge: Understanding the Constraint



Tier-Based Strictly Local Morphology



- Work by Alëna Aksënova, Thomas Graf, and Sophie Moradi.
- It seems that **morphology is also TSL**.
  (Aksënova et al. 2016)
- Morphology $\equiv$ Morphotactics of underlying forms
  but see (Aksënova and De Santo 2017) on derivations
- We are unaware of any non-TSL patterns in this realm.
- Tight typology, explains gaps

Example: Circumfixation in Indonesian

- Indonesian has circumfixation with no upper bound on the distance between the two parts of the circumfix.

(8)

	maha	siswa
	big	pupil

‘student’

(9)

*(ke-)	maha	siswa	*(-an)
NMN-	big	pupil	-NMN

‘student affairs’

- Requirements: exactly one *ke-* and exactly one *-an*

Tier$_1$	contains all NMN affixes	\$	an			ke	ke	\$
Tier$_0$	contains all morphemes	|	|			|	|	|
n-**grams**	**\$an**, **ke\$**, **keke**, **anan**	\$	an	m	s	ke	ke	\$

Example: Circumfixation in Indonesian

- Indonesian has circumfixation with no upper bound on the distance between the two parts of the circumfix.

(8) maha siswa
big pupil
'student'

(9) *(ke-) maha siswa *(-an)
NMN- big pupil -NMN
'student affairs'

- Requirements: exactly one *ke-* and exactly one *-an*

Tier$_1$	contains all NMN affixes	$	an			ke	ke	$
Tier$_0$	contains all morphemes	|	|			|	|	|
n**-grams**	**$an**, **ke$**, **keke**, **anan**	$	an	m	s	ke	ke	$

Example: Swahili *vyo*

Swahili *vyo* is **either a prefix or a suffix**,
depending on presence of negation. (?)

(10) a.

a-	vi-	**soma**	**-vyo**
SBJ:CL.1-	OBJ:CL.8-	read	-REL:CL.8

‘reads’

b.

a-	si-	**vyo-**	vi-	**soma**
SBJ:CL.1-	NEG-	REL:CL.8-	read	-OBJ:CL.8

‘doesn’t read’

Example: Swahili *vyo* [cont.]

(11) a. * a- **vyo-** vi- **soma**
SBJ:CL.1- REL:CL.8- OBJ:CL.8- read

b. * a- **vyo-** vi- **soma** **-vyo**
SBJ:CL.1- REL:CL.8- OBJ:CL.8- read -REL:CL.8

c. * a- si- **vyo-** vi- **soma** **-vyo**
SBJ:CL.1- NEG- REL:CL.8- OBJ:CL.8- read REL:CL.8-

d. * a- si- vi- **soma** **-vyo**
SBJ:CL.1- NEG- OBJ:CL.8- read REL:CL.8-

Example: Swahili *vyo* [cont.]

Generalizations About *vyo*

- may occur at most once
- must follow negation prefix *si-* if present
- is a prefix iff *si-* is present

Tier$_1$	contains *vyo*, *si*, and stem edges #	
Tier$_0$	contains all morphemes	
n-**grams**	**vyovyo**, **vyo##vyo**	"at most one *vyo*"
	vyosi, **vyo##si**	"*vyo* follows *si*"
	si##vyo, **$vyo##**	"*vyo* is prefix iff *si* present"

Explaining Typological Gaps

Restriction to TSL can also explain some typological gaps.

General Strategy

- Attested patterns **A** and **B** are TSL.
- But combined pattern **A**+**B** is not attested.
- Show that **A**+**B** is not TSL.

Example: Compounding Markers

- Russian has an infix *-o-* that may occur between parts of compounds.
- Turkish has a single suffix *-sı* that occurs at end of compounds.

(12)	vod	**-o-**	voz	**-o-**	voz	
	water	-COMP-	carry	-COMP-	carry	
	'carrier of water-carriers'					
(13)	türk	bahçe	kapı	**-sı**	(*-**sı**)	
	turkish	garden	gate	-COMP	(*-COMP)	
	'Turkish garden gate'					

- **New Universal**
  If a language allows unboundedly many compound affixes, they are **infixes**.

Example: Compounding Markers [cont.]

- Russian and Turkish are TSL.

Tier_1	COMP affix and stem edges #
Russian n-grams	**oo**, **$o**, **o$**
Turkish n-grams	**sisi**, **$si**, **si#**

- The combined pattern would yield Ruskish: $\text{stem}^{n+1}\text{-si}^{n}$
- This pattern is not regular and hence **not TSL either**.

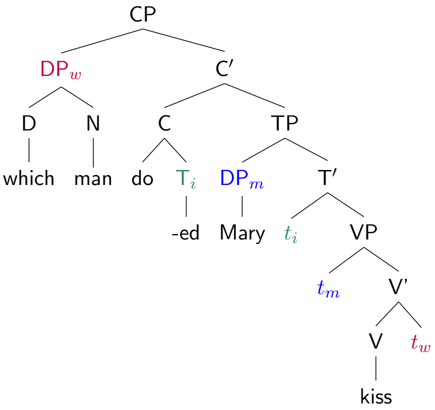
Interim Summary: Morphology

- While we know less about morphology than phonology at this point, it also seems to be TSL.
- Even complex patterns like Swahili *vyo* can be captured.
- At the same time, we get **new universals**:

Bounded Circumfixation
: No recursive process can be realized via circumfixation.

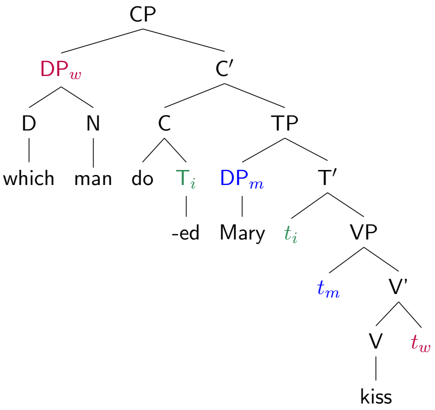
- We can reuse tools and techniques from TSL phonology, including learning algorithms.
- The cognitive resource requirements are also comparable.

MGs & Derivation Trees

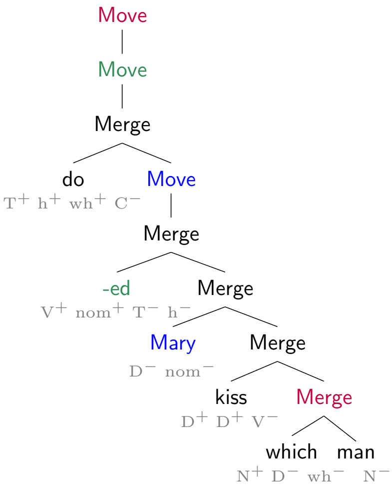


Phrase Structure Tree

MGs & Derivation Trees



Phrase Structure Tree



Derivation Tree

Constraints on Move

What about Move?

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

Constraints on Movement

Move	Every head with a negative Move feature is dominated by a matching Move node.
SMC	Every Move node is a closest dominating match for exactly one head.

Constraints on Move

What about Move?

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

Constraints on Movement

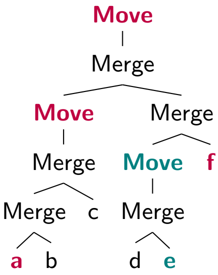
Move	Every head with a negative Move feature is dominated by a matching Move node.
SMC	Every Move node is a closest dominating match for exactly one head.

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 local**.
- What if every movement type (wh, topic, ...) induces its own tier? Would that make Move dependencies local?

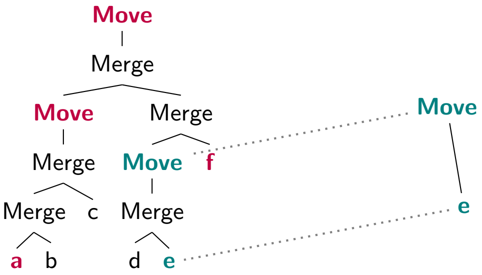
Tiers for Movement

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



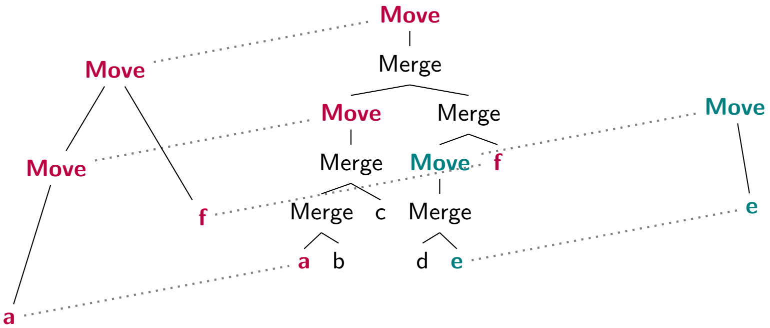
Tiers for Movement

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

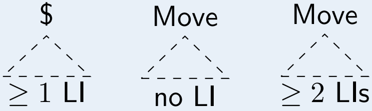


Move Constraints over Tiers

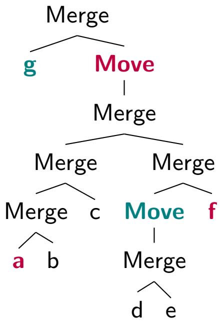
	Original	Tier
Move	Every head with a negative Move feature is dominated by a matching Move node.	Every lexical item has a **mother** labeled Move.
SMC	Every Move node is a closest dominating match for exactly one head.	Exactly one of a Move node's **daughters** is a lexical item.

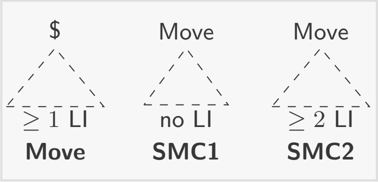
Tree n-gram Templates

Move **SMC1** **SMC2**

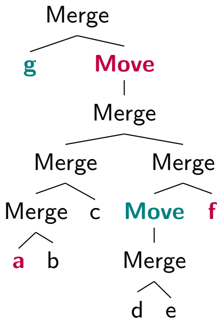


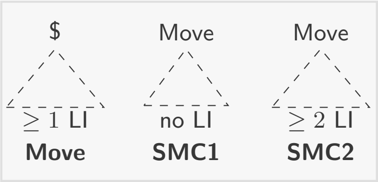
Example of Ill-Formed Derivation



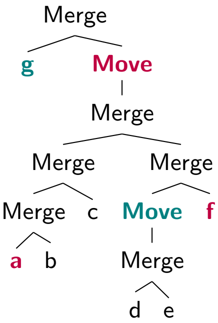


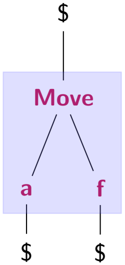
Example of Ill-Formed Derivation

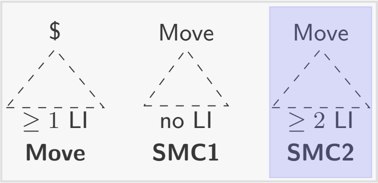




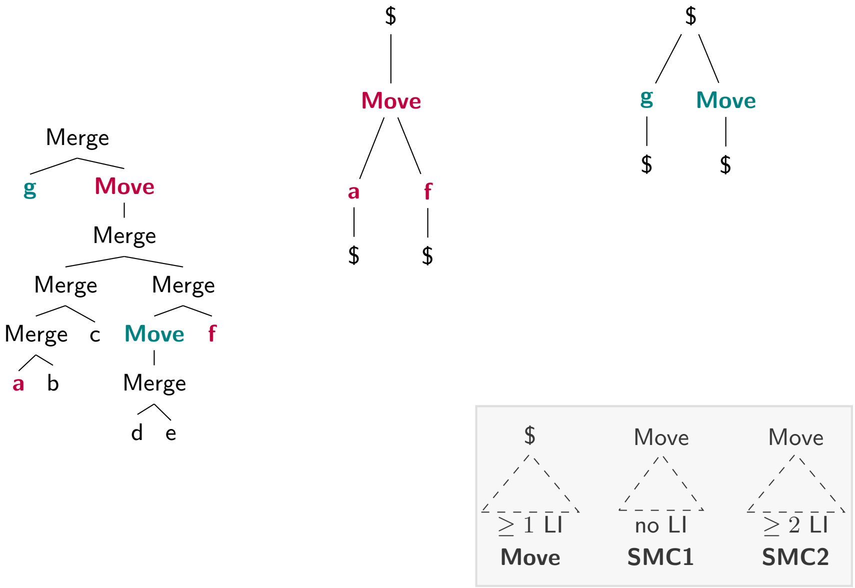
Example of Ill-Formed Derivation



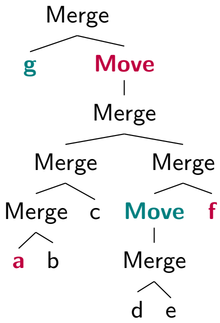


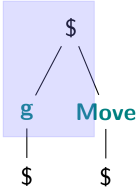


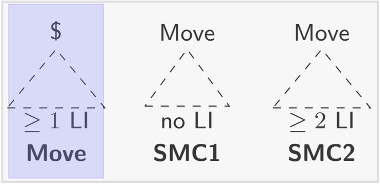
Example of Ill-Formed Derivation



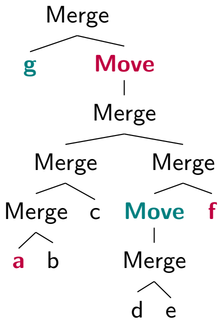
Example of Ill-Formed Derivation

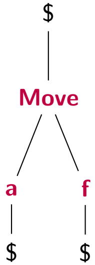


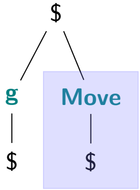


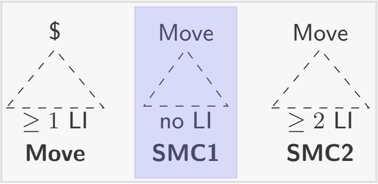


Example of Ill-Formed Derivation

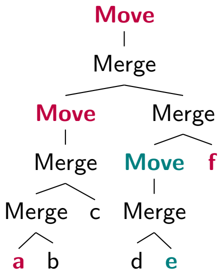


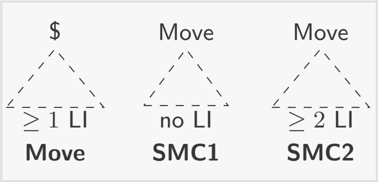




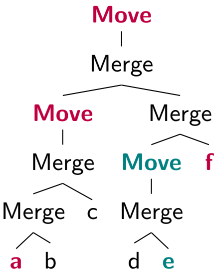


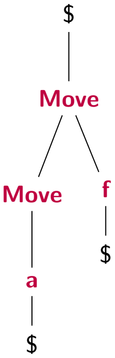
Example of Well-Formed Derivation

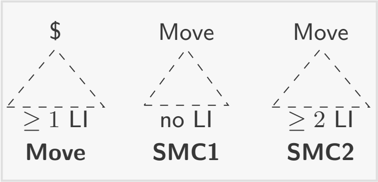




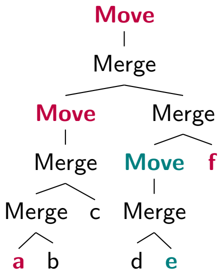
Example of Well-Formed Derivation

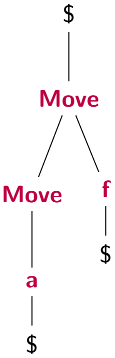


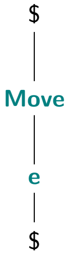


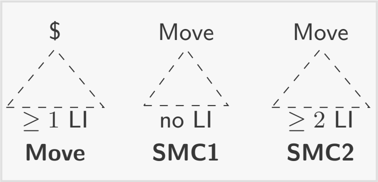


Example of Well-Formed Derivation









Remarks on Single Movement Normal Form

- Single Movement Normal Form seems unrealistic.
- **But:** does not rule out multiple movement steps, only says there is **single feature trigger in derivation**
- Intermediate landing sites can be part of structure built from the derivation tree.

A Conjecture on Movement Restrictions (Graf 2017a)

- Conversion of an MG into single movement normal form causes large blow-up in size of lexicon.
- Blow-up varies a lot: from 0 to hundred times the original size
- The more fixed the position of movers, the smaller the blow-up $\Rightarrow$ island constraints as a means to limit lexical blow-up?

Remarks on Single Movement Normal Form

- Single Movement Normal Form seems unrealistic.
- **But:** does not rule out multiple movement steps, only says there is **single feature trigger in derivation**
- Intermediate landing sites can be part of structure built from the derivation tree.

A Conjecture on Movement Restrictions (Graf 2017a)

- Conversion of an MG into single movement normal form causes large blow-up in size of lexicon.
- Blow-up varies a lot: from 0 to hundred times the original size
- The more fixed the position of movers, the smaller the blow-up $\Rightarrow$ island constraints as a means to limit lexical blow-up?

The Central Role of Derivation Trees

- Derivation trees are rarely considered in generative syntax. (but see Epstein et al. 1998)
- satisfy Chomsky's structural desiderata:
  - no linear order
  - label-free
  - extension condition
  - inclusiveness condition
- contain all information to produce phrase structure trees
  ⇒ **central data structure** of Minimalist syntax

Psychological Reality of Derivation Trees

Central role of derivation trees backed up by **processing data**:

- Derivation trees can be parsed top-down (Stabler 2013)
- Parsing models update Derivational Theory of Complexity, make correct processing predictions for
  - right $<$ center embedding (Kobele et al. 2012)
  - crossing $<$ nested dependencies (Kobele et al. 2012)
  - SC-RC $<$ RC-SC (**?**)
  - SRC $<$ ORC in English (**?**)
  - SRC $<$ ORC in East-Asian (**?**)
  - quantifier scope preferences (Pasternak 2016)

Technical Fertility of Derivation Trees

Derivation trees made it easy for MGs to accommodate the full syntactic toolbox:

- sidewards movement (Stabler 2006; Graf 2013)
- affix hopping (Graf 2012b, 2013)
- clustering movement (Gärtner and Michaelis 2010)
- tucking in (Graf 2013)
- ATB movement (Kobele 2008)
- copy movement (Kobele 2006)
- extraposition (Hunter and Frank 2014)
- Late Merge (Kobele 2010; Graf 2014a)
- Agree (Kobele 2011; Graf 2012a)
- adjunction (Fowlie 2013; Graf 2014b; Hunter 2015)
- TAG-style adjunction (Graf 2012c)

Samala (Revisited)

Sibilant Harmony in Samala (McMullin 2016)

1) Unbounded sibilant harmony

a. /k-su-ʃojin/	kʃuʃojin	"I darken it"
b. /k-su-k'ili-mekeken-ʃ/	kʃuk'ilimekeketʃ	"I straighten up"

2) /s/→ [ʃ] when preceding (adjacent) [t, n, l]

a. /s-lok'in/	ʃlok'in	"he cuts it"
b. /s-tepuʔ/	ʃtepuʔ	"he gambles"

3) Long-distance agreement overrides local disagreement

a. /s-iʃt-iʃti-jep-us/	sististijepus	"they show him"
b. /s-net-us/	snetus	"he does it to him"

Samala (Revisited)

Sibilant Harmony in SAMALA (McMullin 2016)

1) Unbounded sibilant harmony

a.	/k-su-ʃojin/	kʃuʃojin	"I darken it"
b.	/k-su-k'ili-mekeken-ʃ/	kʃuk'ilimekeketʃ	"I straighten up"

2) /s/→ [ʃ] when preceding (adjacent) [t, n, l]

a.	/s-lok'in/	ʃlok'in	"he cuts it"
b.	/s-tepuʔ/	ʃtepuʔ	"he gambles"

3) Long-distance agreement overrides local disagreement

a.	/s-iʃt-iʃti-jep-us/	sististijepus	"they show him"
b.	/s-net-us/	snetus	"he does it to him"

Samala (Revisited)

Sibilant Harmony in SAMALA (McMullin 2016)

1) Unbounded sibilant harmony

a.	/k-su-ʃojin/	kʃuʃojin	"I darken it"
b.	/k-su-k'ili-mekeken-ʃ/	kʃuk'ilimekeketʃ	"I straighten up"

2) /s/→ [ʃ] when preceding (adjacent) [t, n, l]

a.	/s-lok'in/	ʃlok'in	"he cuts it"
b.	/s-tepuʔ/	ʃtepuʔ	"he gambles"

3) Long-distance agreement overrides local disagreement

a.	/s-iʃt-iʃti-jep-us/	sististijepus	"they show him"
b.	/s-net-us/	snetus	"he does it to him"

Structure-Sensitive TSL (SS-TSL)

Samala Sibilant Harmony (Revisited)

- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization

s n e t u s

Structure-Sensitive TSL (SS-TSL)

Samala Sibilant Harmony (Revisited)

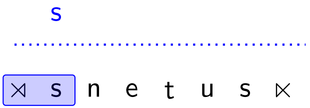
- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization

⋊ s n e t u s ⋉

Structure-Sensitive TSL (SS-TSL)

SAMALA Sibilant Harmony (Revisited)

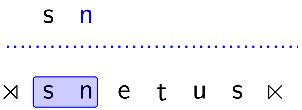
- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization



Structure-Sensitive TSL (SS-TSL)

Samala Sibilant Harmony (Revisited)

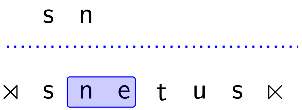
- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization



Structure-Sensitive TSL (SS-TSL)

Samala Sibilant Harmony (Revisited)

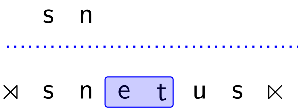
- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization



Structure-Sensitive TSL (SS-TSL)

Samala Sibilant Harmony (Revisited)

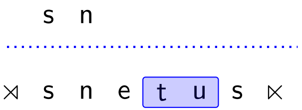
- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization



Structure-Sensitive TSL (SS-TSL)

Samala Sibilant Harmony (Revisited)

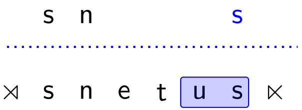
- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization



Structure-Sensitive TSL (SS-TSL)

Samala Sibilant Harmony (Revisited)

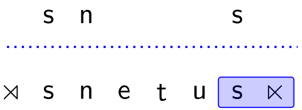
- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization



Structure-Sensitive TSL (SS-TSL)

Samala Sibilant Harmony (Revisited)

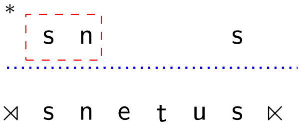
- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization



Structure-Sensitive TSL (SS-TSL)

SAMALA Sibilant Harmony (Revisited)

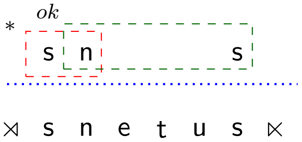
- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization



Structure-Sensitive TSL (SS-TSL)

Samala Sibilant Harmony (Revisited)

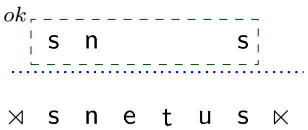
- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization



Structure-Sensitive TSL (SS-TSL)

SAMALA Sibilant Harmony (Revisited)

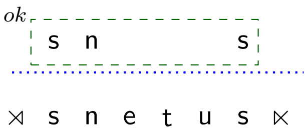
- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization



Structure-Sensitive TSL (SS-TSL)

SAMALA Sibilant Harmony (Revisited)

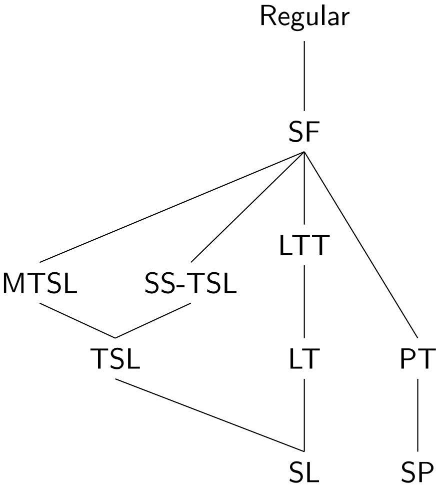
- anticipatory sibilant harmony
- palatalization to avoid local restrictions
- sibilant harmony overrides palatalization



Grammar

T $= \{\ \sigma : \sigma \in \{$s, ʃ$\} \vee (\sigma \in \{$ n, t, l $\} \wedge$ s $\prec^{+} \sigma)\}$
S $= \{$*sʃ, *sʃ, *sn(¬s), *st(¬s), *sl(¬s)$\}$

SS -TSL: Relations to other Classes



The TSL Neighborhood: a Plethora of Combinations