Fragments of First-Order Logic for Linguistic Structures

Thomas Graf

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

UConn Logic Colloquium
Apr 19, 2017

The Talk in a Nutshell

Narrow Goal

Find the smallest fragment of first-order logic that is sufficiently expressive for natural language structures.

Why it Matters

- discover new parallels between phonology, morphology, syntax, even semantics
- explain typological gaps
- new empirical predictions
- simplify the learning problem
- benefit NLP applications

The Talk in a Nutshell

Narrow Goal

Find the smallest fragment of first-order logic that is sufficiently expressive for natural language structures.

Why it Matters

- discover new parallels between phonology, morphology, syntax, even semantics
- explain typological gaps
- new empirical predictions
- simplify the learning problem
- benefit NLP applications

Bigger Picture for...

- **Semanticists/Philosophers of Language**
  - Logic is not limited to natural language meaning.
  - It is just as useful for studying natural language structures.
- **Linguists in General**
  - Mathematical abstraction is a good thing.
  - It captures insights that are usually lost among the details.
- **Logicians**
  - Very weak logics are very relevant.
  - There are tons of problems to be solved.

Outline

1. Logics for Phonology
   - Logic and Linguistic Structures
   - Application to Phonology
   - TSL: Relativized Precedence
2. Beyond TSL Phonology
   - TSL Morphology
   - TSL Morpho-Semantics
   - TSL Syntax
3. Open Problems
   - Better Formal Understanding of TSL
   - Mappings Between Structures

Linguistic Structures

Linguists distinguish three major levels of structure:

1. **Phonology** = sound structure

word-final devoicing	rad $\rightarrow$ rat
primary stress	axiom

2. **Morphology** = word structure

inflection	she run+s
derivation	in+decipher+able

3. **Syntax** = sentence structure

John does not like Mary.
*Not John does like Mary.

Linguistic Structures

Linguists distinguish three major levels of structure:

1. **Phonology** = sound structure

word-final devoicing	rad → rat
primary stress	ˈaks.ɹ.əm, not ˈaks.ɹ.ˈəm

2. **Morphology** = word structure

inflection	she run+s
derivation	in+decipher+able

3. **Syntax** = sentence structure

   John does not like Mary.

   *Not John does like Mary.

The Big Linguistic Questions

- What are the laws that govern each structural level?
- How **complex** are these laws? How hard are they to compute?
- Do we find **typological gaps**, i.e. patterns that should exist but don't appear in any language?
- What can we infer about human cognition?

The Computational Program

- Computer scientists have figured out a lot about complexity, so let's apply their ideas to language.
- Formal language theory and **logic** greatly deepen our understanding of language.

The Big Linguistic Questions

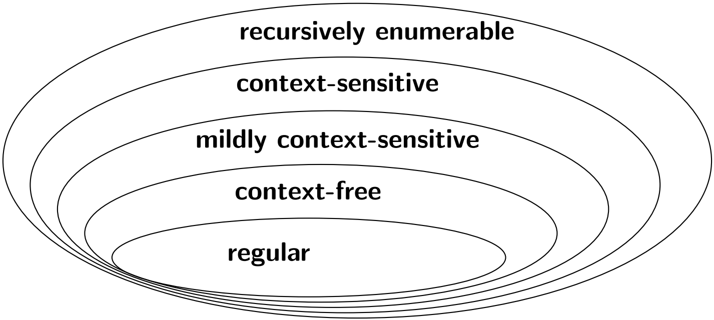
- What are the laws that govern each structural level?
- How **complex** are these laws? How hard are they to compute?
- Do we find **typological gaps**, i.e.
  patterns that should exist but don't appear in any language?
- What can we infer about human cognition?

The Computational Program

- Computer scientists have figured out a lot about complexity, so let's apply their ideas to language.
- Formal language theory and **logic** greatly deepen our understanding of language.

A Familiar Picture: The Chomsky Hierarchy

- The perceivable output of language is strings (sequences of sound waves, words, sentences).
- The complexity of string languages is measured by the (extended) **Chomsky hierarchy**. (Chomsky 1956, 1959)



A Different Picture: The Subregular Hierarchy

Often forgotten: hierarchy of **subregular languages**
(McNaughton and Papert 1971; Rogers et al. 2010)

$$
\begin{array}{ccc}
 & & \text{Regular} \\
 & & \cup \\
\text{Locally Threshold Testable} & \subset & \text{Star Free} \\
\cup & & \cup \\
\text{Locally Testable} & & \text{Piecewise Testable} \\
\cup & & \cup \\
\text{Strictly Local} & & \text{Strictly Piecewise}
\end{array}
$$

A Different Picture: The Subregular Hierarchy

Often forgotten: hierarchy of **subregular languages**
(McNaughton and Papert 1971; Rogers et al. 2010)

		Regular	Monadic Second-Order Logic
		$\cup$	
Locally Threshold Testable	$\subset$	Star Free	
$\cup$		$\cup$	
Locally Testable		Piecewise Testable	
$\cup$		$\cup$	
Strictly Local		Strictly Piecewise	

A Different Picture: The Subregular Hierarchy

Often forgotten: hierarchy of **subregular languages**
(McNaughton and Papert 1971; Rogers et al. 2010)

		Regular	Monadic Second-Order Logic
		∪	
Locally Threshold Testable	⊂	Star Free	First-Order Logic
∪		∪	
Locally Testable		Piecewise Testable	
∪		∪	
Strictly Local		Strictly Piecewise	

A Different Picture: The Subregular Hierarchy

Often forgotten: hierarchy of **subregular languages**
(McNaughton and Papert 1971; Rogers et al. 2010)

		Regular	Monadic Second-Order Logic
		$\cup$	
Locally Threshold Testable	$\subset$	Star Free	First-Order Logic
$\cup$		$\cup$	
Locally Testable		Piecewise Testable	Propositional Logic
$\cup$		$\cup$	
Strictly Local		Strictly Piecewise	

A Different Picture: The Subregular Hierarchy

Often forgotten: hierarchy of **subregular languages**
(McNaughton and Papert 1971; Rogers et al. 2010)

		Regular	Monadic Second-Order Logic
		$\cup$	
Locally Threshold Testable	$\subset$	Star Free	First-Order Logic
$\cup$		$\cup$	
Locally Testable		Piecewise Testable	Propositional Logic
$\cup$		$\cup$	
Strictly Local		Strictly Piecewise	Conjunction of Negative Literals

A Different Picture: The Subregular Hierarchy

Often forgotten: hierarchy of **subregular languages**
(McNaughton and Papert 1971; Rogers et al. 2010)

		Regular	Monadic Second-Order Logic
		$\cup$	
Locally Threshold Testable	$\subset$	Star Free	First-Order Logic
$\cup$		$\cup$	
Locally Testable		Piecewise Testable	Propositional Logic
$\cup$		$\cup$	
Strictly Local		Strictly Piecewise	Conjunction of Negative Literals
$S/\triangleleft$			

A Different Picture: The Subregular Hierarchy

Often forgotten: hierarchy of **subregular languages**
(McNaughton and Papert 1971; Rogers et al. 2010)

	Regular	Monadic Second-Order Logic
	$\cup$	
Locally Threshold Testable $\subset$	Star Free	First-Order Logic
$\cup$	$\cup$	
Locally Testable	Piecewise Testable	Propositional Logic
$\cup$	$\cup$	
Strictly Local	Strictly Piecewise	Conjunction of Negative Literals
$S/\triangleleft$	$</\triangleleft^+$	

Example: Word-Final Devoicing is SL

- Captured by forbidding voiced segments at the end of a word
- **German**: Don't have **z$** or **v$** (where **$** = word edge).

Corresponding Logical Formula with $\lhd$

CNL	Modal	FO
$\neg$**z$**	$\neg(\mathbf{z} \wedge \lhd \mathbf{\$})$	$\neg(\exists x, y[\mathbf{z}(x) \wedge \mathbf{\$}(y) \wedge x \lhd y])$
$\wedge$	$\wedge$	$\wedge$
$\neg$**v$**	$\neg(\mathbf{v} \wedge \lhd \mathbf{\$})$	$\neg(\exists x, y[\mathbf{v}(x) \wedge \mathbf{\$}(y) \wedge x \lhd y])$

Example

alalas	$a ∧ al ∧ la ∧ as ∧ s$
*alalaz	$a ∧ al ∧ la ∧ as ∧ z$

Example: Word-Final Devoicing is SL

- Captured by forbidding voiced segments at the end of a word
- **German**: Don't have **z$** or **v$** (where **$** = word edge).

Corresponding Logical Formula with $\triangleleft$

CNL	Modal	FO
¬z$	$\neg(\mathbf{z} \wedge \triangleleft\$)$	$\neg(\exists x, y[\mathbf{z}(x) \wedge \$(y) \wedge x \triangleleft y])$
$\wedge$	$\wedge$	$\wedge$
¬v$	$\neg(\mathbf{v} \wedge \triangleleft\$)$	$\neg(\exists x, y[\mathbf{v}(x) \wedge \$(y) \wedge x \triangleleft y])$

Example

alalas	\$a ∧ al ∧ la ∧ as ∧ s\$
*alalaz	\$a ∧ al ∧ la ∧ as ∧ z\$

Example: Word-Final Devoicing is SL

- Captured by forbidding voiced segments at the end of a word
- **German**: Don't have **z$** or **v$** (where **$** = word edge).

Corresponding Logical Formula with $\triangleleft$

CNL	Modal	FO
$\neg$z$	$\neg(\mathbf{z} \wedge \triangleleft \$)$	$\neg(\exists x, y[\mathbf{z}(x) \wedge \$(y) \wedge x \triangleleft y])$
$\wedge$	$\wedge$	$\wedge$
$\neg$v$	$\neg(\mathbf{v} \wedge \triangleleft \$)$	$\neg(\exists x, y[\mathbf{v}(x) \wedge \$(y) \wedge x \triangleleft y])$

Example

alalas	$a ∧ al ∧ la ∧ as ∧ s$
*alalaz	$a ∧ al ∧ la ∧ as ∧ z$

Example: Intervocalic Voicing is SL

- Captured by forbidding voiceless segments between vowels
- **Suppose**:
  - $[-\text{voice}] = \{$s,ʃ$\}$
  - $\text{V} = \{$a,i,u$\}$
- **Then**: don't have **asa**, **aʃa**, **asi**, **aʃi**, ...

Corresponding Logical Formula with $\triangleleft$

CNL	**Modal**	**FO**
¬**asa**	$\neg(\mathsf{a} \wedge \triangleleft \mathsf{s} \wedge \triangleleft \triangleleft \mathsf{a})$	$\neg(\exists x, y, z[\mathsf{a}(x) \wedge \mathsf{s}(y) \wedge \mathsf{a}(z) \wedge x \triangleleft y \wedge y \triangleleft z])$
$\wedge$	$\wedge$	$\wedge$
¬**aʃa**	$\neg(\mathsf{a} \wedge \triangleleft \mathsf{ʃ} \wedge \triangleleft \triangleleft \mathsf{a})$	$\neg(\exists x, y, z[\mathsf{a}(x) \wedge \mathsf{ʃ}(y) \wedge \mathsf{a}(z) \wedge x \triangleleft y \wedge y \triangleleft z])$
$\wedge$	$\wedge$	$\wedge$
...	...	...

Example: Sibilant Voicing Harmony is SP

- If multiple sibilants (s,z,ʃ,ʒ) occur in the same word, they must all be voiceless (s,ʃ) or voiced (z,ʒ).
- In other words: Don't mix **purple** and **teal**.
- **But:** Sibilants can be arbitrarily far away from each other!

Corresponding Logical Formula with $\triangleleft^+$

CNL	Modal	FO
¬zs	$\neg(\mathsf{z} \wedge \triangleleft^+ \mathsf{s})$	$\neg(\exists x, y[\mathsf{z}(x) \wedge \mathsf{s}(y) \wedge x \triangleleft^+ y])$
$\wedge$	$\wedge$	$\wedge$
¬zʃ	$\neg(\mathsf{z} \wedge \triangleleft^+ \mathsf{ʃ})$	$\neg(\exists x, y[\mathsf{z}(x) \wedge \mathsf{ʃ}(y) \wedge x \triangleleft^+ y])$
$\wedge$	$\wedge$	$\wedge$
...	...	...

Example

salalas	\$s ∧ \$a ∧ \$l ∧ \$\$ ∧ sa ∧ sl ∧ ss ∧ s\$ ∧ al ∧ aa ∧ ...
*zalalas	\$z ∧ \$a ∧ \$l ∧ \$s ∧ \$\$ ∧ za ∧ zl ∧ zs ∧ z\$ ∧ al ∧ aa ∧ ...

Example: Sibilant Voicing Harmony is SP

- If multiple sibilants (s,z,ʃ,ʒ) occur in the same word, they must all be voiceless (s,ʃ) or voiced (z,ʒ).
- In other words: Don't mix **purple** and **teal**.
- **But:** Sibilants can be arbitrarily far away from each other!

Corresponding Logical Formula with $\lhd^+$

CNL	**Modal**	**FO**
¬**zs**	$\neg(\mathbf{z} \wedge \lhd^+ \mathbf{s})$	$\neg(\exists x, y[\mathbf{z}(x) \wedge \mathbf{s}(y) \wedge x \lhd^+ y])$
$\wedge$	$\wedge$	$\wedge$
¬**zʃ**	$\neg(\mathbf{z} \wedge \lhd^+ \text{ʃ})$	$\neg(\exists x, y[\mathbf{z}(x) \wedge \text{ʃ}(y) \wedge x \lhd^+ y])$
$\wedge$	$\wedge$	$\wedge$
...	...	...

Example

salalas	\$s ∧ \$a ∧ \$l ∧ \$\$ ∧ sa ∧ sl ∧ ss ∧ s\$ ∧ al ∧ aa ∧ ...
*zalalas	\$z ∧ \$a ∧ \$l ∧ \$s ∧ \$\$ ∧ za ∧ zl ∧ zs ∧ z\$ ∧ al ∧ aa ∧ ...

Example: Sibilant Voicing Harmony is SP

- If multiple sibilants (**s**,**z**,**ʃ**,**ʒ**) occur in the same word, they must all be voiceless (**s**,**ʃ**) or voiced (**z**,**ʒ**).
- In other words: Don't mix **purple** and **teal**.
- **But:** Sibilants can be arbitrarily far away from each other!

Corresponding Logical Formula with $\triangleleft^+$

CNL	**Modal**	**FO**
¬**zs**	$\neg(\mathbf{z} \wedge \triangleleft^+ \mathbf{s})$	$\neg(\exists x, y[\mathbf{z}(x) \wedge \mathbf{s}(y) \wedge x \triangleleft^+ y])$
∧	∧	∧
¬**zʃ**	$\neg(\mathbf{z} \wedge \triangleleft^+ \mathbf{ʃ})$	$\neg(\exists x, y[\mathbf{z}(x) \wedge \mathbf{ʃ}(y) \wedge x \triangleleft^+ y])$
∧	∧	∧
...	...	...

Example

salalas	\$s ∧ \$a ∧ \$l ∧ \$\$ ∧ sa ∧ sl ∧ ss ∧ s\$ ∧ al ∧ aa ∧ ...
*zalalas	\$z ∧ \$a ∧ \$l ∧ \$s ∧ \$\$ ∧ za ∧ zl ∧ zs ∧ z\$ ∧ al ∧ aa ∧ ...

Example: Primary Stress is LTT

- Every word has exactly one primary stress.
- SL, SP, and LT are too weak:

	SL	**SP**	**LT**
at most one stress	no	yes	no
at least one stress	no	no	yes

  - SL fails because this is a non-local dependency.
  - SP fails because it can only forbid presence, not absence.
  - LT only distinguishes "exactly 0" and "strictly more than 0".
- We need LTT, i.e. FO with successor.

Corresponding Logical Formula (no relation required)

$$\exists! x [\bigvee_{\alpha \text{ a segment with primary stress}} \alpha(x)]$$

Problem 1: LTT is Too Powerful

First-order logic can combine restrictions too freely
$\Rightarrow$ **massive overgeneration**

	$\forall \mathbf{V}, \forall \mathbf{S}$
If there is a vowel **V** at the end, and	$\Big[\bigvee_{\alpha \text{ a vowel}} \alpha(\mathbf{V}) \wedge \exists y [\$(y) \wedge \mathbf{V} \triangleleft y] \wedge$
there is a sibilant **S** such that	$\big(\mathsf{s}(\mathbf{S}) \vee \mathsf{z}(\mathbf{S}) \vee \int(\mathbf{S}) \vee \mathsf{ʒ}(\mathbf{S})\big) \wedge$
either **S** is word-initial or	$\neg\Big(\exists x [\$(x) \wedge x \triangleleft \mathbf{S}] \leftrightarrow$
there are exactly seven consonants,	$\exists!^7 x_i \big[\bigwedge_{1 \leq i \leq 7} \big(\bigvee_{\alpha \text{ a consonant}} \alpha(x_i)\big)\big]\Big)$
then intervocalic voicing is enforced iff	$\rightarrow \Big((\neg \mathsf{VsV} \wedge \neg \mathsf{V\int V}) \leftrightarrow$
the voicing value of **S** is	$\big(([-\text{voice}](\mathbf{S}) \wedge [+\text{round}](\mathbf{V})) \vee$
the opposite of **V**'s value for round	$([+\text{voice}](\mathbf{S}) \wedge [-\text{round}](\mathbf{V}))\big)\Big)\Big]$

Problem 1: LTT is Too Powerful

First-order logic can combine restrictions too freely

$\Rightarrow$ **massive overgeneration**

	$\forall \mathbf{V}, \forall \mathbf{S}$
If there is a vowel **V** at the end, and	$\Big[\bigvee_{\alpha \text{ a vowel}} \alpha(\mathbf{V}) \wedge \exists y[\$(y) \wedge \mathbf{V} \triangleleft y] \wedge$
there is a sibilant **S** such that	$\big(\text{s}(\mathbf{S}) \vee \text{z}(\mathbf{S}) \vee \text{ʃ}(\mathbf{S}) \vee \text{ʒ}(\mathbf{S})\big) \wedge$
either **S** is word-initial or	$\neg\Big(\exists x[\$(x) \wedge x \triangleleft \mathbf{S}] \leftrightarrow$
there are exactly seven consonants,	$\exists!^7 x_i \big[\bigwedge_{1 \leq i \leq 7} \big(\bigvee_{\alpha \text{ a consonant}} \alpha(x_i)\big)\big]\Big)$
then intervocalic voicing is enforced iff	$\rightarrow \Big(\big(\neg \text{VsV} \wedge \neg \text{VʃV}\big) \leftrightarrow$
the voicing value of **S** is	$\big(([-\text{voice}](\mathbf{S}) \wedge [+\text{round}](\mathbf{V})) \vee$
the opposite of **V**'s value for round	$([+\text{voice}](\mathbf{S}) \wedge [-\text{round}](\mathbf{V}))\big)\Big)\Big]$

Problem 2: Scattered Distribution

Phonological dependencies seem to be **weirdly distributed**.

		Regular	Monadic Second-Order Logic
		$\cup$	
Locally Threshold Testable	$\subset$	Star Free	First-Order Logic
$\cup$		$\cup$	
Locally Testable		Piecewise Testable	Propositional Logic
$\cup$		$\cup$	
Strictly Local		Strictly Piecewise	Conjunction of Negative Literals
$S/\triangleleft$		$< / \triangleleft^+$	

Problem 2: Scattered Distribution

Phonological dependencies seem to be **weirdly distributed**.

		Regular	Monadic Second-Order Logic
		$\cup$	
Locally Threshold Testable	$\subset$	Star Free	First-Order Logic
$\cup$		$\cup$	
Locally Testable		Piecewise Testable	Propositional Logic
$\cup$		$\cup$	
Strictly Local	$\subset$ TSL	Strictly Piecewise	Conjunction of Negative Literals
$S/\lhd$		$< / \lhd^{+}$	

A New Challenger: Tier-Based Strictly Local

- First defined in Heinz et al. (2011)
- TSL is a minimal expansion of SL.
- TSL replaces $\triangleleft$ by a relativized version of $\triangleleft^+$, denoted $\triangleleft_{\mathbf{T}}$.
- Inspired by **phonological tiers**.

Jeff Heinz

Defining Tier-Precedence $\triangleleft_{\mathbf{T}}$

Given alphabet Σ, a tier is some $\mathbf{T} \subseteq \Sigma$.

$$\mathbf{x} \triangleleft_{\mathbf{T}} \mathbf{y} \Leftrightarrow \mathbf{T}(\mathbf{x}) \wedge \mathbf{T}(\mathbf{y}) \wedge \mathbf{x} \triangleleft^+ \mathbf{y} \wedge \neg\exists \mathbf{z}[\mathbf{T}(\mathbf{z}) \wedge \mathbf{x} \triangleleft^+ \mathbf{z} \wedge \mathbf{z} \triangleleft^+ \mathbf{y}]$$

$$\mathbf{T}(\mathbf{x}) \Leftrightarrow \bigvee_{t \in \mathbf{T}} t(\mathbf{x})$$

Example: Sibilant Voicing Harmony Revisited

- Reminder: Don't mix **purple** and **teal**.
- $\mathsf{T} := \{\mathsf{s}, \mathsf{z}, \mathsf{ʃ}, \mathsf{ʒ}\}$
- Don't allow **zs**, **zʃ**, **sz**, **sʒ**, ...

$ s s $

$ a s e l a s i $

$ z s $

$ a z e l a s i $

Corresponding Logical Formula with $\triangleleft_\mathsf{T}$

CNL	Modal	FO
$\neg$**zs**	$\neg(\mathsf{z} \wedge \triangleleft_\mathsf{T} \mathsf{s})$	$\neg(\exists x, y[\mathsf{z}(x) \wedge \mathsf{s}(y) \wedge x \triangleleft_\mathsf{T} y])$
$\wedge$	$\wedge$	$\wedge$
$\neg$**zʃ**	$\neg(\mathsf{z} \wedge \triangleleft_\mathsf{T} \mathsf{ʃ})$	$\neg(\exists x, y[\mathsf{z}(x) \wedge \mathsf{ʃ}(y) \wedge x \triangleleft_\mathsf{T} y])$
$\wedge$	$\wedge$	$\wedge$
...	...	...

Example: Sibilant Voicing Harmony Revisited

- Reminder: Don't mix **purple** and **teal**.
- $T := \{\mathsf{s}, \mathsf{z}, \int, \mathsf{ʒ}\}$
- Don't allow **zs**, **zʃ**, **sz**, **sʒ**, ...

```
$   s         s   $            $   z         s   $
|   |         |   |            |   |         |   |
$ a s e l a s i $              $ a z e l a s i $
```

Corresponding Logical Formula with $\triangleleft_T$

CNL	Modal	FO
$\neg$zs	$\neg(\mathsf{z} \wedge \triangleleft_T \mathsf{s})$	$\neg(\exists x, y[\mathsf{z}(x) \wedge \mathsf{s}(y) \wedge x \triangleleft_T y])$
$\wedge$	$\wedge$	$\wedge$
$\neg$zʃ	$\neg(\mathsf{z} \wedge \triangleleft_T ʃ)$	$\neg(\exists x, y[\mathsf{z}(x) \wedge ʃ(y) \wedge x \triangleleft_T y])$
$\wedge$	$\wedge$	$\wedge$
...	...	...

Example: Primary Stress Revisited

- Every word has exactly one primary stress.
- We don't need LTT, **TSL is sufficient**:

	SL	**SP**	**LT**	**LTT**	**TSL**
at most one stress	no	yes	no	yes	yes
at least one stress	no	no	yes	yes	yes

- **T** contains all stressed segments $\acute{\sigma}$.
- **At most one stress**: don't have $\acute{\sigma}\acute{\sigma}$
- **At least one stress**: don't have $$

$				$	$	á		á	$	$	á			$
|				|	|	|		|	|	|	|			|
$	a	l	a	$	$	á	l	á	$	$	á	l	a	$

A Single Locus in the Subregular Hierarchy

Phonological dependencies now neatly fit into
a **contiguous subregion** of the subregular hierarchy.

			Regular	Monadic Second-Order Logic
			$\cup$	
Locally Threshold Testable		$\subset$	Star Free	First-Order Logic
$\cup$			$\cup$	
Locally Testable			Piecewise Testable	Propositional Logic
$\cup$			$\cup$	
Strictly Local	$\subset$	TSL	Strictly Piecewise	Conjunction of Negative Literals
$S/\triangleleft$			$< / \triangleleft^+$	

Outline

Going Beyond Phonology

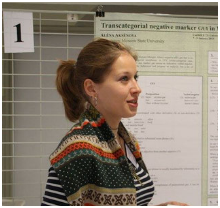
TSL provides a good fit for phonological dependencies.

The $\$10^6$ Question

Is TSL also a good fit for other linguistic structures?

- Morphology?
- Syntax?

TSL Morphology



Alëna Aksënova

Sophie Moradi

- Joint work with Alëna Aksënova and Sophie Moradi.
- It seems that **morphology is also TSL**.
  (Aksënova et al. 2016)

Example: Circumfixation in Indonesian

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

(1) maha siswa
big pupil
'student'

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

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

T	contains all NMN affixes
forbidden	an, ke, keke, anan

$ an ke ke $

$ 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.

(1) maha siswa
big pupil
'student'

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

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

T	contains all NMN affixes
forbidden	**$an**, **ke$**, **keke**, **anan**

$	an			ke	ke	$
|	|			|	|	|
$	an	m	s	ke	ke	$

Explaining a Typological Gap

- In general, affixation can be unbounded.

morgen	tomorrow
über+morgen	the day after tomorrow
(über+)^{n}morgen	(the day after)^n tomorrow

- This pattern is SL and hence TSL.
- But **circumfixation cannot be unbounded** (e.g. Ilocano).

bigát	tomorrow
ka+bigát+an	the day after tomorrow
* *ka+ka+bigát+an+an*	the day after the day after tomorrow

Explanation

- The pattern would be *ka*^{n}*+bigát+an*^{n}.
- This is not first-order definable and hence not TSL.

Explaining a Typological Gap

- In general, affixation can be unbounded.

morgen	tomorrow
über+morgen	the day after tomorrow
(über+)^n morgen	(the day after)^n tomorrow

- This pattern is SL and hence TSL.
- But **circumfixation cannot be unbounded** (e.g. Ilocano).

bigát	tomorrow
ka+bigát+an	the day after tomorrow
* *ka+ka+bigát+an+an*	the day after the day after tomorrow

Explanation

- The pattern would be *ka^n+bigát+an^n*.
- This is not first-order definable and hence not TSL.

Explaining a Typological Gap

- In general, affixation can be unbounded.

morgen	tomorrow
über+morgen	the day after tomorrow
(über+)^n morgen	(the day after)^n tomorrow

- This pattern is SL and hence TSL.
- But **circumfixation cannot be unbounded** (e.g. Ilocano).

bigát	tomorrow
ka+bigát+an	the day after tomorrow
* *ka+ka+bigát+an+an*	the day after the day after tomorrow

Explanation

- The pattern would be *ka*^{n}*+bigát+an*^{n}.
- This is not first-order definable and hence not TSL.

TSL Morpho-Semantics?

The importance of TSL for word structure seems to extend even into semantics.

Case Study: Generalized Quantifiers (Graf 2017b)

A generalized quantifier may have a monomorphemic realization only if its quantifier language is TSL.

- Let's take this step by step:
  - Monomorphemic?
  - Generalized quantifier?
  - Quantifier language?

TSL Morpho-Semantics?

The importance of TSL for word structure seems to extend even into semantics.

Case Study: Generalized Quantifiers (Graf 2017b)

A generalized quantifier may have a monomorphemic realization only if its quantifier language is TSL.

- Let's take this step by step:
  - Monomorphemic?
  - Generalized quantifier?
  - Quantifier language?

Typology of Generalized Quantifiers

Monomorphemic not assembled from smaller parts

Quantifier	Can be monomorphemic?
every	yes
no	yes
some	yes
not all	no
two	yes
all but one	no
an even number	no
a third of	no
most	???

Reminder: Generalized Quantifiers

Generalized quantifier **Q**(**A**, **B**):

- two sets **A** and **B** as arguments
- returns truth value $(0, 1)$

Example

(3) Every student cheated.

- **every**(**A**, **B**) $= 1$ iff **A** $\subseteq$ **B**
- **student**: John, Mary, Sue
- **cheat**: John, Mary
- **student** $\not\subseteq$ **cheat** $\Rightarrow$ **every**(**student**, **cheat**) $= 0$
- "Every student cheated" is false.

Binary Strings

- The language of **A** is the set of all permutations of **A**.

Example

student	John, Mary, Sue
$L($**student**$)$	John Mary Sue, John Sue Mary
	Mary John Sue, Mary Sue John
	Sue John Mary, Sue Mary John

- Now replace every **a** $\in$ **A** by a truth value:
  1 if **a** $\in$ **B**
  0 if **a** $\notin$ **B**
- The result is the **binary string language** of **A** under **B**.

Example

student	John, Mary, Sue
cheat	John, Mary
binary strings	110, 101, 011

Binary Strings

- The language of **A** is the set of all permutations of **A**.

Example

student	John, Mary, Sue
L(**student**)	John Mary Sue, John Sue Mary
	Mary John Sue, Mary Sue John
	Sue John Mary, Sue Mary John

- Now replace every **a** $\in$ **A** by a truth value:
  1 if **a** $\in$ **B**
  0 if **a** $\notin$ **B**
- The result is the **binary string language** of **A** under **B**.

Example

student	John, Mary, Sue
cheat	John, Mary
binary strings	110, 101, 011

Quantifier Languages

- Quantifier accepts only binary strings of specific shape
- This is its **quantifier language**.

Example: *every*

- **every**(**A**, **B**) holds iff **A** $\subseteq$ **B**
- So every element of **A** must be mapped to 1.
- All strings must be sequences of 1.
- $L(\textbf{every}) = \{1\}^*$

Example: *some*

- **some**(**A**, **B**) holds iff **A** $\cap$ **B** $\neq \emptyset$
- Some element of **A** must be mapped to 1.
- $L(\textbf{some}) = \{0,1\}^* \, 1 \, \{0,1\}^*$

Quantifier Languages

- Quantifier accepts only binary strings of specific shape
- This is its **quantifier language**.

Example: *every*

- **every**(**A**, **B**) holds iff **A** $\subseteq$ **B**
- So every element of **A** must be mapped to 1.
- All strings must be sequences of 1.
- $L($**every**$) = \{1\}^*$

Example: *some*

- **some**(**A**, **B**) holds iff **A** $\cap$ **B** $\neq \emptyset$
- Some element of **A** must be mapped to 1.
- $L($**some**$) = \{0,1\}^* \, 1 \, \{0,1\}^*$

Quantifier Languages

- Quantifier accepts only binary strings of specific shape
- This is its **quantifier language**.

Example: *every*

- **every**(**A**, **B**) holds iff **A** $\subseteq$ **B**
- So every element of **A** must be mapped to 1.
- All strings must be sequences of 1.
- $L(\textbf{every}) = \{1\}^*$

Example: *some*

- **some**(**A**, **B**) holds iff **A** $\cap$ **B** $\neq \emptyset$
- Some element of **A** must be mapped to 1.
- $L(\textbf{some}) = \{0,1\}^* \, 1 \, \{0,1\}^*$

Overview of Quantifier Languages

If a quantifier language is not TSL,
then its quantifier cannot be monomorphemic in any language.

Quantifier	**Constraint**	**TSL Description**	**Mono.**
every	no 0	**T** $:= \{0,1\}$, $\neg 0$	yes
no	no 1	**T** $:= \{0,1\}$, $\neg 1$	yes
some	one or more 1	**T** $:= \{1\}$, $\neg \$\$$	yes
not all	one or more 0	**T** $:= \{0\}$, $\neg \$\$$	no
(at least) two	two or more 1	**T** $:= \{1\}$, $\neg \$\$ \wedge \neg \$1\$$	yes
(at most) two	two or fewer 1	**T** $:= \{1\}$, $\neg 111$	yes
all but one	exactly one 0	**T** $:= \{0\}$, $\neg \$\$ \wedge \neg 00$	no
even number	even 1	impossible	no
most	more 1 than 0	impossible	???

Two Important Remarks

- There is good semantic evidence that "most" is internally complex and hence not monomorphemic. (Hackl 2009)
- If we stipulate that $0 \in \mathbf{T}$ implies $1 \in \mathbf{T}$, only monomorphemic quantifiers are left!

Quantifier	Constraint	TSL Description	Mono.
every	no 0	$\mathbf{T} := \{0, 1\}, \neg 0$	yes
no	no 1	$\mathbf{T} := \{0, 1\}, \neg 1$	yes
some	one or more 1	$\mathbf{T} := \{1\}, \neg \$\$$	yes
not all	one or more 0	$\mathbf{T} := \{0\}, \neg \$\$$	no
(at least) two	two or more 1	$\mathbf{T} := \{1\}, \neg \$\$ \wedge \neg \$1\$$	yes
(at most) two	two or fewer 1	$\mathbf{T} := \{1\}, \neg 111$	yes
all but one	exactly one 0	$\mathbf{T} := \{0\}, \neg \$\$ \wedge \neg 00$	no
even number	even 1	impossible	no
most	more 1 than 0	impossible	no

Two Important Remarks

- There is good semantic evidence that "most" is internally complex and hence not monomorphemic. (Hackl 2009)
- If we stipulate that $0 \in$ **T** implies $1 \in$ **T**, only monomorphemic quantifiers are left!

Quantifier	Constraint	TSL Description	Mono.
every	no 0	**T** $:= \{0, 1\}$, $\neg 0$	yes
no	no 1	**T** $:= \{0, 1\}$, $\neg 1$	yes
some	one or more 1	**T** $:= \{1\}$, $\neg\$\$$	yes
not all	one or more 0	**T** $:= \{\mathbf{0}\}$, $\neg\$\$$	**no**
(at least) two	two or more 1	**T** $:= \{1\}$, $\neg\$\$ \wedge \neg\$1\$$	yes
(at most) two	two or fewer 1	**T** $:= \{1\}$, $\neg 111$	yes
all but one	exactly one 0	**T** $:= \{\mathbf{0}\}$, $\neg\$\$ \wedge \neg 00$	**no**
even number	even 1	**impossible**	**no**
most	more 1 than 0	**impossible**	**no**

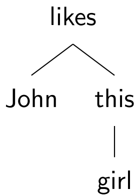
TSL Syntax

- Every sentence hides a very elaborate **tree structure**.
- Linguists assume two structural notions:

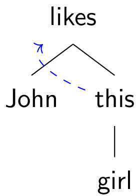
  Dependency encodes functor-argument relations ($\approx$ semantics)

  Move displaces subtrees ($\approx$ word order)

(4) John likes this girl.



(5) This girl, John likes.



- In addition, all movements are triggered by **features**.

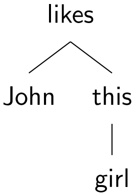
TSL Syntax

- Every sentence hides a very elaborate **tree structure**.
- Linguists assume two structural notions:

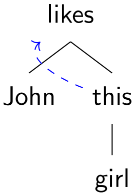
  Dependency encodes functor-argument relations (≈ semantics)

  Move displaces subtrees (≈ word order)

(4) John likes this girl.



(5) This girl, John likes.



- In addition, all movements are triggered by **features**.

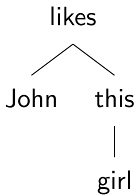
TSL Syntax

- Every sentence hides a very elaborate **tree structure**.
- Linguists assume two structural notions:

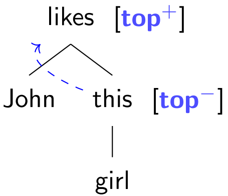
  Dependency encodes functor-argument relations ($\approx$ semantics)

  Move displaces subtrees ($\approx$ word order)

(4) John likes this girl.



(5) This girl, John likes.



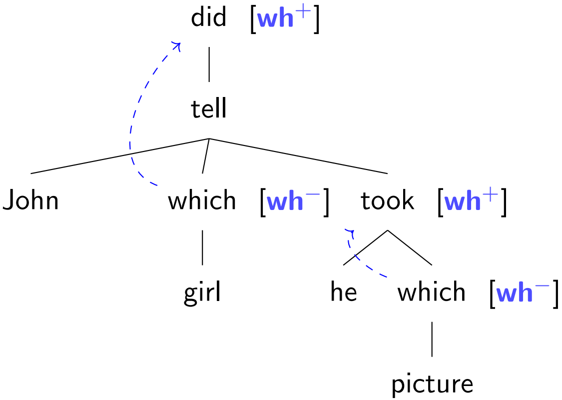
- In addition, all movements are triggered by **features**.

More Complex Movement Configurations

A single sentence can contain multiple movements.

Movers always target the **closest matching node**.

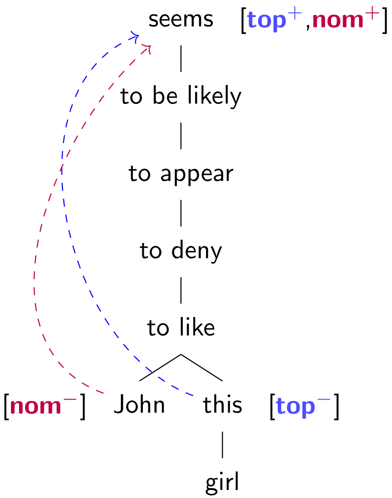
(6) Which girl did John tell which picture he took.



Unbounded Movement

The length of movement steps is **unbounded** (◁ won't be enough).

(7) This girl, John seems to be likely to appear to deny to like.



Exact Matching

And feature polarities must line up **one-to-one**.

(8) John was attacked.

(9) * Was attacked John.

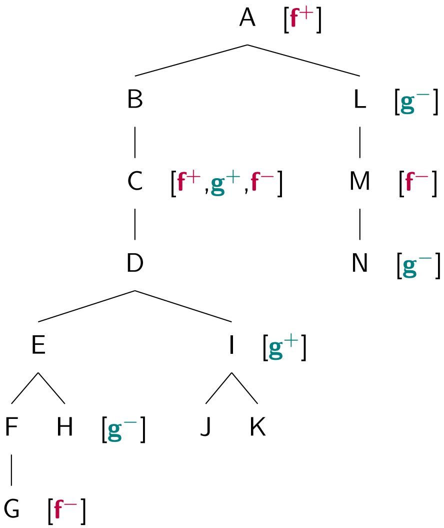
was attacked [**nom**$^{+}$]
|
John [**nom**$^{-}$]

was attacked
|
John [**nom**$^{-}$]

was attacked [**nom**$^{+}$]
|
John

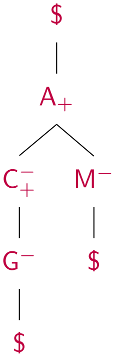
An Abstract Example

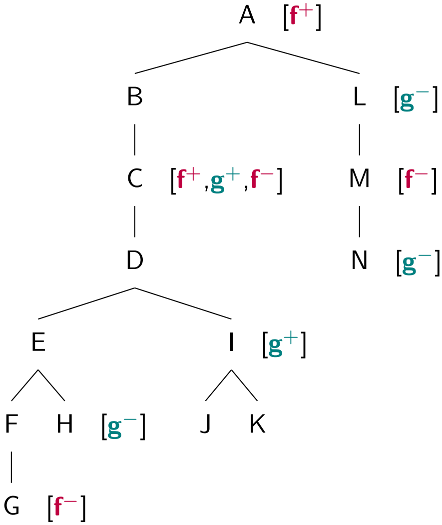
Is the tree below well-formed? We can **solve it with TSL!**



An Abstract Example

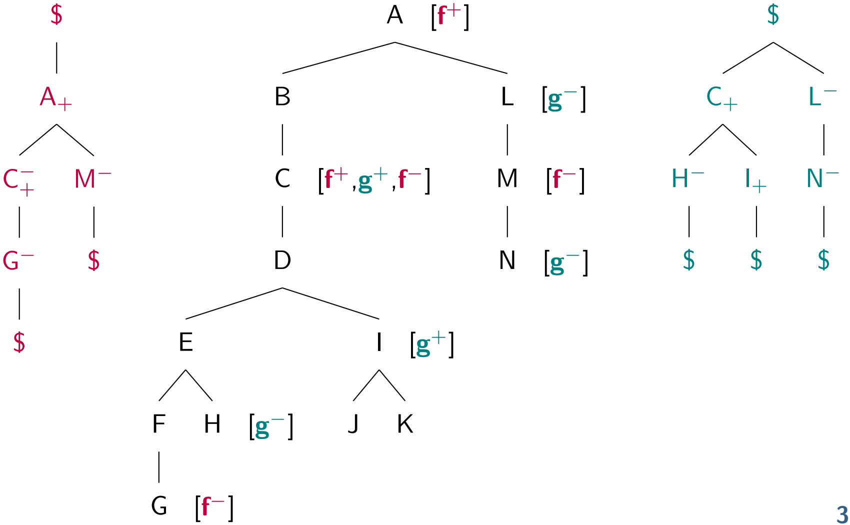
Is the tree below well-formed? We can **solve it with TSL!**





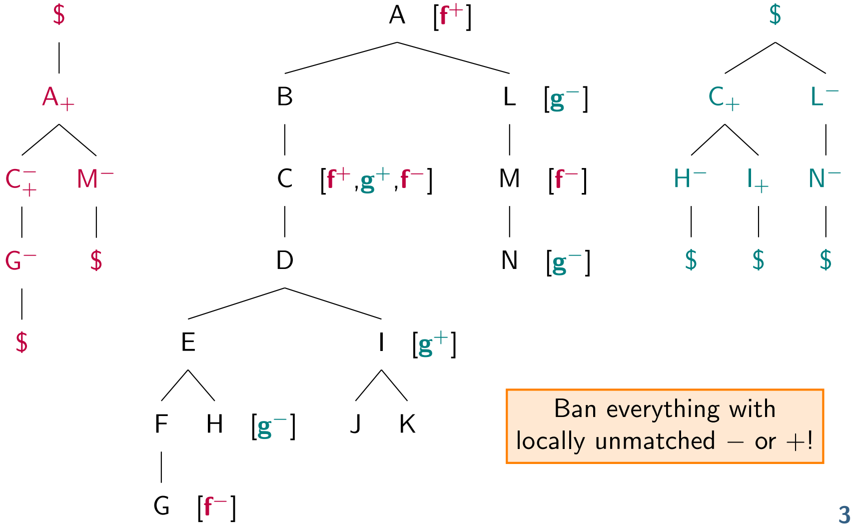
An Abstract Example

Is the tree below well-formed? We can **solve it with TSL!**



An Abstract Example

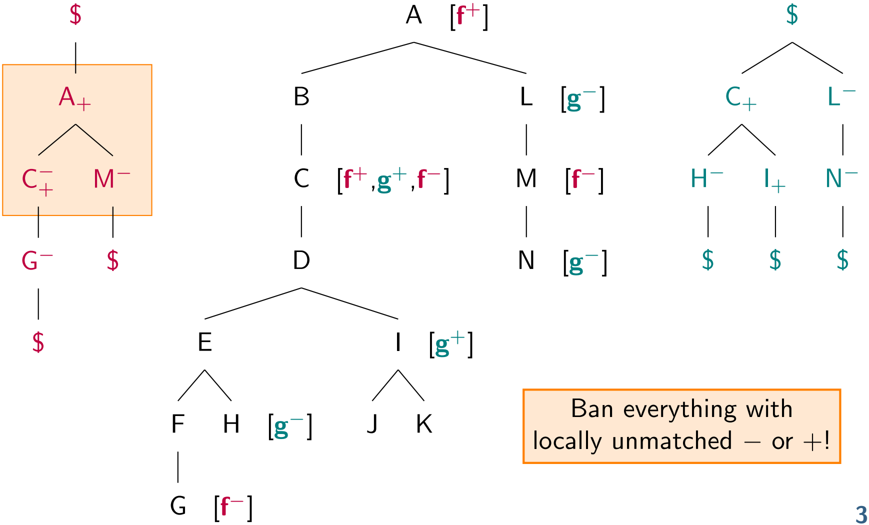
Is the tree below well-formed? We can **solve it with TSL!**



Ban everything with locally unmatched − or +!

An Abstract Example

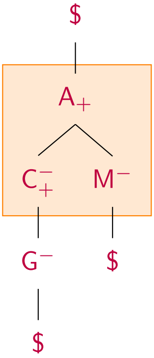
Is the tree below well-formed? We can **solve it with TSL!**

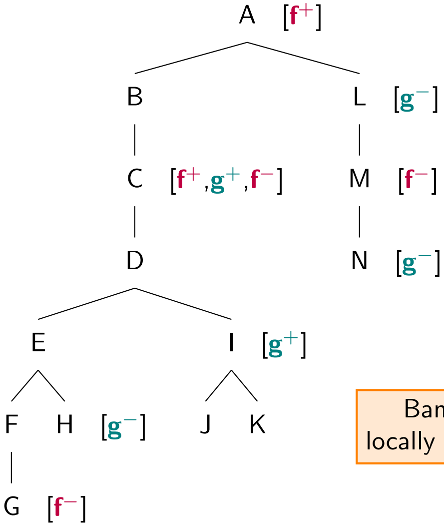


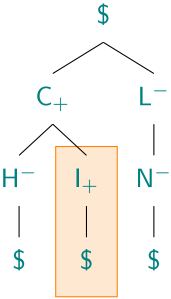
Ban everything with locally unmatched − or +!

An Abstract Example

Is the tree below well-formed? We can **solve it with TSL!**



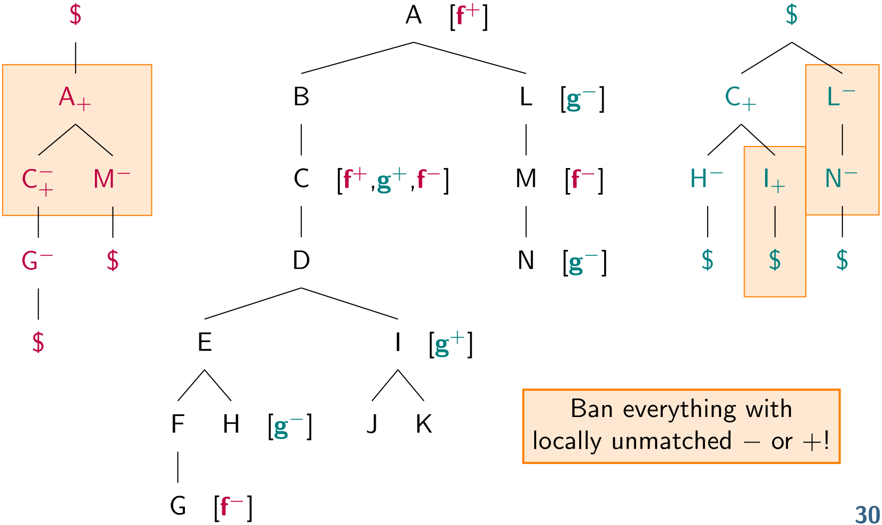




Ban everything with locally unmatched $-$ or $+$!

An Abstract Example

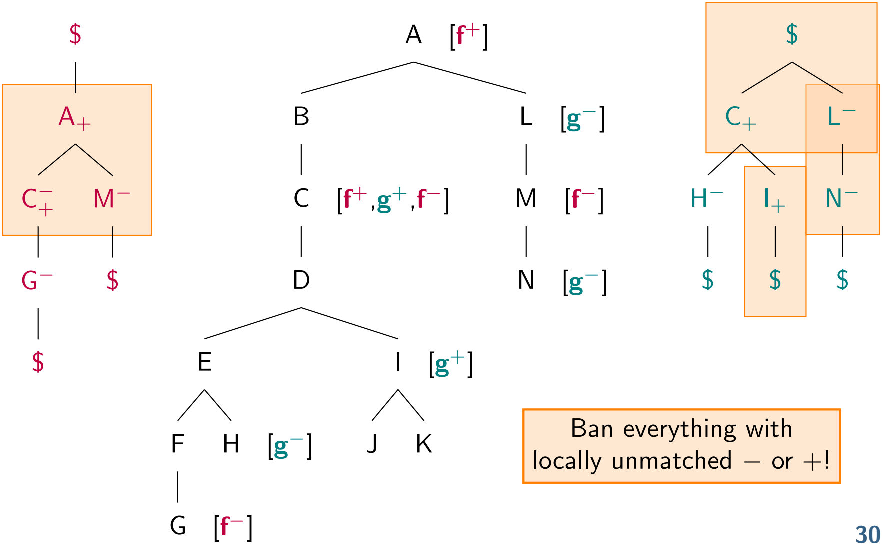
Is the tree below well-formed? We can **solve it with TSL!**



Ban everything with locally unmatched − or +!

An Abstract Example

Is the tree below well-formed? We can **solve it with TSL!**



Ban everything with locally unmatched − or +!

Why Syntax is TSL

- Syntactic structures encode
  - head-argument dependencies,
  - movement dependencies.
- Movement is controlled by a precise feature calculus.
- Given a tree, we can easily project a “tree tier” for each type of movement feature.
- Those tiers greatly reduce the complexity of the problem: **movement dependencies hold between adjacent nodes**.
- Hence we can block illicit local configurations as usual.

Interim Summary

Logical View of TSL

TSL = Conjunction of Negative Literals with $\triangleleft_{T}$

- Phonology and morphology only have TSL dependencies (with a few exceptions).
- TSL plays a role even in morpho-semantics.
- The core of syntax (dependencies, movement) is TSL, too.

Strong Parallelism Hypothesis

All linguistic structures only involve dependencies that are TSL (or at most a minor extension of TSL).

Interim Summary

Logical View of TSL

TSL = Conjunction of Negative Literals with $\triangleleft_T$

- Phonology and morphology only have TSL dependencies (with a few exceptions).
- TSL plays a role even in morpho-semantics.
- The core of syntax (dependencies, movement) is TSL, too.

Strong Parallelism Hypothesis

All linguistic structures only involve dependencies that are TSL (or at most a minor extension of TSL).

Outline

The Open Problems

1. We do not understand TSL well.
2. TSL undergenerates slightly.
3. We also need mappings between structures.

Open Questions About TSL

- Right now, dependencies are shown to be TSL by providing a formula/grammar. A more abstract technique would be much more efficient.
- Non-TSL is cumbersome to prove:
  - consider all possible tiers
  - show that none work

What We Need

- pumping lemma
- decomposition theorems (if $L \notin X \cap Y$, then $L \notin \mathrm{TSL}$)
- smallest-counterexample results
- ...

TSL Undergeneration

- There are a few, very rare phenomena that require slightly more expressivity than TSL:
  1. local blocking of unbounded sibilant harmony
  2. RHOL-like stress patterns
  3. unbounded tone plateauing
  4. unbounded circumambient processes
- (1) and (2) are handled with minimal extensions of TSL. (Baek 2017; De Santo 2017; De Santo and Graf 2017)
- (3) and (4) require an extension of SP:
  FO formulas of the form $\Delta \rightarrow \Phi$, where
  - Δ is a **domain formula**
  - Φ is an SP formula (= CNL formula with $\triangleleft^+$)

  (Graf 2016)

Mappings and Transductions

- Linguists care deeply about **mappings between structures**.

Example: Word-Final Devoicing

- It is not enough that voiced consonants are forbidden at the end of a word.
- There are principled alternations that need to be captured:

[raːt]	*wheel* or *advice*			
[raːtə]	*advice*.Pl	$\Leftarrow$	/raːt/	*advice*
[reːdə]	*wheel*.Pl		/raːd/	*wheel*

- Logical models of mappings (= transductions) exist, but are too powerful.

A First-Order Transduction

- A logical transduction operates by representing **one structure inside another**.

Example: String Reversal as an FO transduction

Input Relation: $\lhd$

Output Relation: $\blacktriangleleft$ $\qquad x \blacktriangleleft y \Leftrightarrow y \lhd x$

$$0 \xrightarrow{\lhd} 1 \xrightarrow{\lhd} 2 \xrightarrow{\lhd} 3 \xrightarrow{\lhd} 4 \xrightarrow{\lhd} 5$$

- No natural language mapping is capable of reversal.
- Like in the case of LTT, full FO is too much.

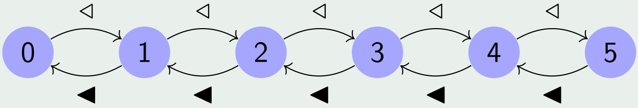
A First-Order Transduction

- A logical transduction operates by representing **one structure inside another**.

Example: String Reversal as an FO transduction

Input Relation: $\triangleleft$

Output Relation: $\blacktriangleleft$ $\quad x \blacktriangleleft y \Leftrightarrow y \triangleleft x$



- No natural language mapping is capable of reversal.
- Like in the case of LTT, full FO is too much.

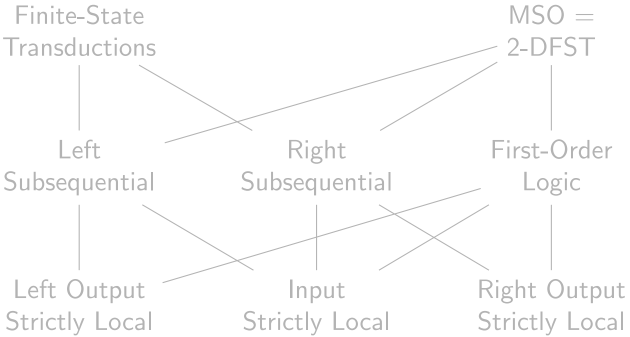
Important Questions About Transductions

1. What are linguistically reasonable fragments of FO for transductions?
2. Are they closed under composition? (A linguistic grammar usually is a sequence of mappings.)
3. Do they preserve definability in TSL/FO/MSO?
4. What is the strongest class of transductions that preserves TSL?

Snapshot of Ongoing Research

Jane Chandlee

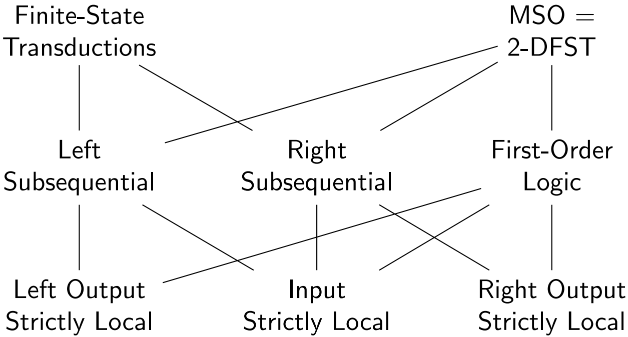
Weak string transductions have been studied, but their **connection to logic is unclear**.
(Mohri 1997; Engelfriet and Hoogeboom 2001; Courcelle and Engelfriet 2012; Chandlee 2014)



Snapshot of Ongoing Research

Weak string transductions have been studied, but their **connection to logic is unclear**.
(Mohri 1997; Engelfriet and Hoogeboom 2001; Courcelle and Engelfriet 2012; Chandlee 2014)

Jane Chandlee



Snapshot of Ongoing Research

Weak string transductions have been studied, but their **connection to logic is unclear**.
(Mohri 1997; Engelfriet and Hoogeboom 2001; Courcelle and Engelfriet 2012; Chandlee 2014)

Jane Chandlee

Finite-State Transductions

MSO = 2-DFST

Left Subsequential

Right Subsequential

First-Order Logic

Left Output Strictly Local

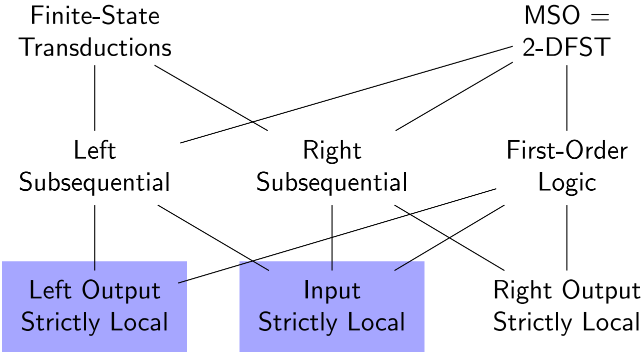
Input Strictly Local

Right Output Strictly Local

Snapshot of Ongoing Research

Weak string transductions have been studied, but their **connection to logic is unclear**. (Mohri 1997; Engelfriet and Hoogeboom 2001; Courcelle and Engelfriet 2012; Chandlee 2014)

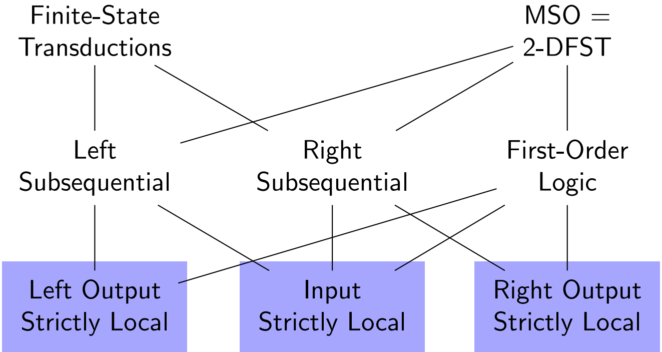
Jane Chandlee



Snapshot of Ongoing Research

Weak string transductions have been studied, but their **connection to logic is unclear**.
(Mohri 1997; Engelfriet and Hoogeboom 2001; Courcelle and Engelfriet 2012; Chandlee 2014)

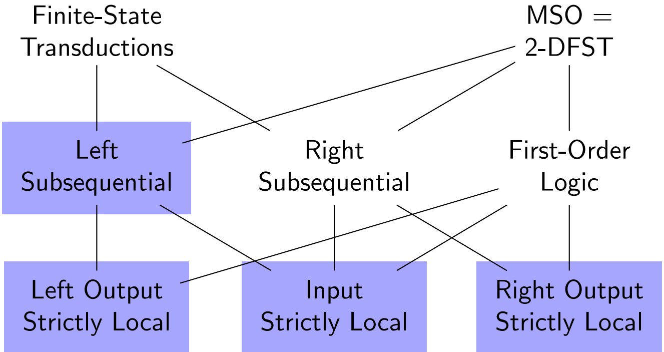
Jane Chandlee



Snapshot of Ongoing Research

Weak string transductions have been studied, but their **connection to logic is unclear**. (Mohri 1997; Engelfriet and Hoogeboom 2001; Courcelle and Engelfriet 2012; Chandlee 2014)

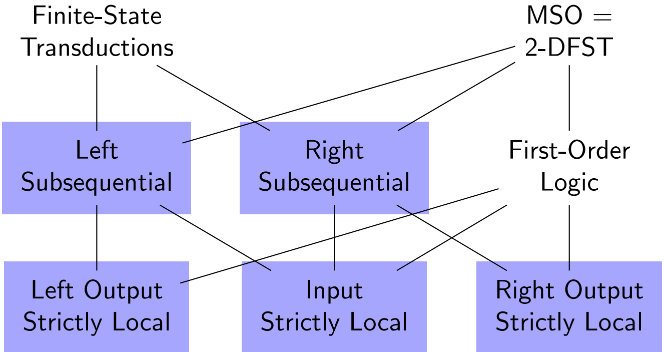
Jane Chandlee



Snapshot of Ongoing Research

Weak string transductions have been studied, but their **connection to logic is unclear**.
(Mohri 1997; Engelfriet and Hoogeboom 2001; Courcelle and Engelfriet 2012; Chandlee 2014)

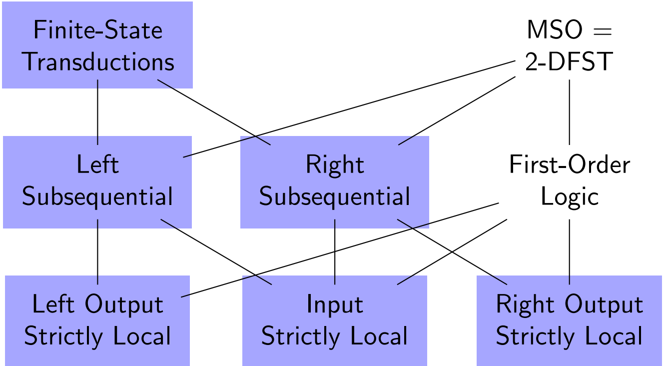
Jane Chandlee



Snapshot of Ongoing Research

Weak string transductions have been studied, but their **connection to logic is unclear**.
(Mohri 1997; Engelfriet and Hoogeboom 2001; Courcelle and Engelfriet 2012; Chandlee 2014)

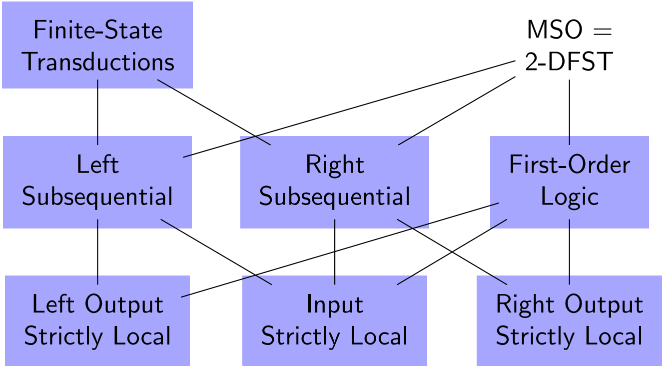
Jane Chandlee



Snapshot of Ongoing Research

Weak string transductions have been studied, but their **connection to logic is unclear**.
(Mohri 1997; Engelfriet and Hoogeboom 2001; Courcelle and Engelfriet 2012; Chandlee 2014)

Jane Chandlee



Conclusion

- When viewed from a logical perspective, language is **surprisingly weak**:
  - Even propositional logic is too much.
  - We need CNL with relativized $\triangleleft^+$.
- This weakness holds **across language modules**.
  - phonology
  - morphology
  - morpho-semantics
  - syntax
- This has major empirical and theoretical implications.
- But many open questions remain.
- In order to address those questions, we will need an **alliance of linguists, logicians, and computer scientists**.

Resources and Readings

1. **Survey papers:** Pullum and Rogers (2006); Heinz (2011a,b, 2015); Rogers and Pullum (2011); Chandlee and Heinz (2016)
2. **TSL and its extensions:** Heinz et al. (2011); McMullin (2016); Baek (2017); De Santo (2017); De Santo and Graf (2017); Graf (2016)
3. **TSL morphology:** Aksënova et al. (2016); Graf (2017a)
4. **TSL morpho-semantics:** Graf (2017b)
5. **TSL syntax:** Graf (2012); Graf and Heinz (2016)
6. **Mappings:** Courcelle and Engelfriet (2012); Chandlee (2014, 2016); Jardine (2016)
7. **Learnability:** Heinz (2010); Kasprzik and Kötzing (2010); Heinz et al. (2012); Jardine et al. (2014); Lai (2015); Jardine and Heinz (2016); Jardine and McMullin (2017)

References I

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

Baek, Hyunah. 2017. Computational representation of unbounded stress: Tiers with structural features. Ms., Stony Brook University.

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

Chandlee, Jane. 2016. Computational locality in morphological maps. Ms., Haverford College.

Chandlee, Jane, and Jeffrey Heinz. 2016. Computational phonology. Ms., Haverford College and University of Delaware.

Chomsky, Noam. 1956. Three models for the description of language. *IRE Transactions on Information Theory* 2:113–124.

Chomsky, Noam. 1959. On certain formal properties of grammars. *Information and Control* 2:137–167.

Courcelle, Bruno, and Joost Engelfriet. 2012. *Graph structure and monadic second-order logic: A language-theoretic approach*. Cambridge, UK: Cambridge University Press.

References II

De Santo, Aniello. 2017. Extending TSL languages: Conjunction as multiple tier-projection. Ms., Stony Brook University.

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

Engelfriet, Joost, and Hendrik Jan Hoogeboom. 2001. MSO definable string transductions and two-way finite-state transducers. *ACM Transactions of Computational Logic* 2:216–254.

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.

Graf, Thomas. 2016. The power of locality domains in phonology. Ms., Stony Brook University.

Graf, Thomas. 2017a. Graph transductions and typological gaps in morphological paradigms. Ms., Stony Brook University.

Graf, Thomas. 2017b. The subregular complexity of monomorphemic quantifiers. Ms., Stony Brook University.

Graf, Thomas, and Jeffrey Heinz. 2016. Tier-based strict locality in phonology and syntax. Ms., Stony Brook University and University of Delaware.

References III

Hackl, Martin. 2009. On the grammar and processing of proportional quantifiers: Most versus more than half. *Natural Language Semantics* 17:63–98.

Heinz, Jeffrey. 2010. String extension learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 897–906. URL http://www.aclweb.org/anthology/P10-1092.pdf.

Heinz, Jeffrey. 2011a. Computational phonology — part I: Foundations. *Language and Linguistics Compass* 5:140–152.

Heinz, Jeffrey. 2011b. Computational phonology — part II: Grammars, learning, and the future. *Language and Linguistics Compass* 5:153–168.

Heinz, Jeffrey. 2015. The computational nature of phonological generalizations. URL http://www.socsci.uci.edu/~lpearl/colareadinggroup/readings/Heinz2015BC_Typology.pdf, ms., University of Delaware.

Heinz, Jeffrey, Anna Kasprzik, and Timo Kötzing. 2012. Learning with lattice-structure hypothesis spaces. *Theoretical Computer Science* 457:111–127.

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.

References IV

Jardine, Adam. 2016. Computationally, tone is different. *Phonology* URL http://udel.edu/~ajardine/files/jardinemscomputationallytoneisdifferent.pdf, to appear.

Jardine, Adam, Jane Chandlee, Rémi Eryaud, and Jeffrey Heinz. 2014. Very efficient learning of structured classes of subsequential functions from positive data. In *Proceedings of the 12th International Conference on Grammatical Inference (ICGI 2014)*, JMLR Workshop Proceedings, 94–108. URL http://www.jmlr.org/proceedings/papers/v34/jardine14a.html.

Jardine, Adam, and Jeffrey Heinz. 2016. Learning tier-based strictly 2-local languages. *Transactions of the ACL* 4:87–98. URL https://aclweb.org/anthology/Q/Q16/Q16-1007.pdf.

Jardine, Adam, and Kevin McMullin. 2017. Efficient learning of tier-based strictly k-local languages. In *Proceedings of Language and Automata Theory and Applications*, Lecture Notes in Computer Science, 64–76. Springer.

Kasprzik, Anna, and Timo Kötzing. 2010. String extension learning using lattices. In *Language and automata theory and applications: 4th international conference, lata 2010, trier, germany, may 24-28, 2010. proceedings*, ed. Adrian-Horia Dediu, Henning Fernau, and Carlos Martín-Vide, 380–391. Berlin, Heidelberg: Springer. URL http://dx.doi.org/10.1007/978-3-642-13089-2_32.

Lai, Regine. 2015. Learnable vs. unlearnable harmony patterns. *Linguistic Inquiry* 46:425–451.

References V

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

McNaughton, Robert, and Seymour Papert. 1971. *Counter-free automata*. Cambridge, MA: MIT Press.

Mohri, Mehryar. 1997. Finite-state transducers in language and speech processing. *Computational Linguistics* 23:269–311.

Pullum, Geoffrey K., and James Rogers. 2006. Animal pattern-learning experiments: Some mathematical background. Ms., Radcliffe Institute for Advanced Study, Harvard University.

Rogers, James, Jeffrey Heinz, Gil Bailey, Matt Edlefsen, Molly Vischer, David Wellcome, and Sean Wibel. 2010. On languages piecewise testable in the strict sense. In *The mathematics of language*, ed. Christan Ebert, Gerhard Jäger, and Jens Michaelis, volume 6149 of *Lecture Notes in Artificial Intelligence*, 255–265. Heidelberg: Springer. URL `http://dx.doi.org/10.1007/978-3-642-14322-9_19`.

Rogers, James, and Geoffrey K. Pullum. 2011. Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information* 20:329–342.