Features:
More trouble than they're worth?

Thomas Graf

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

University of Tromsø
May 27, 2019

You can get the slides here under "News"

The talk in a nutshell

- Evaluation of features from a **computational perspective**
- Very different view
  - not building blocks or properties
  - units of information that mediate computation
- High-level
  - Don't expect much math (but feel free to ask).
  - I'll gloss over many (computational and linguistic) details.

Take-home message

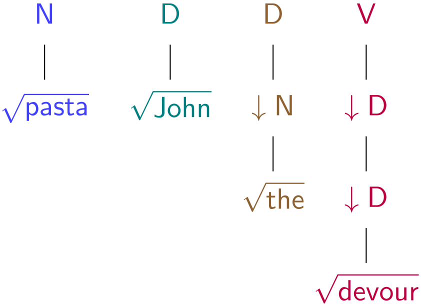
- Features furnish a tremendous amount of power.
- We lack good techniques for limiting them.
- We should prioritize constraints over features.
- Doing so can be linguistically insightful.

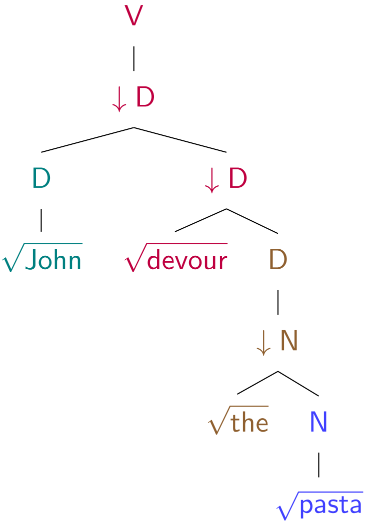
Outline

1. Features ≡ constraints
   - Even 2 features is too much
   - All features are redundant
2. Limiting expressivity
   - Via features: Failure
   - Via constraints: Success
3. Case studies
   - Successive cyclic movement
   - Categories and selection
   - Morphosyntax

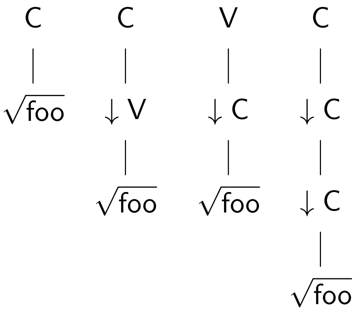
A simple grammar model with subcategorization

Lexicon finite set of **feature treelets**



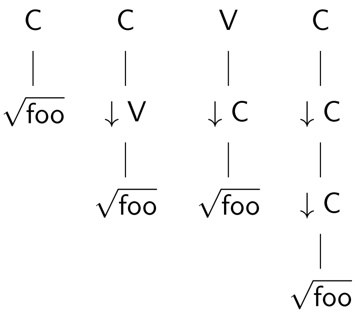


Subcategorization overgenerates with only 2 features



- What happens if every sentence must be a CP?

Subcategorization overgenerates with only 2 features



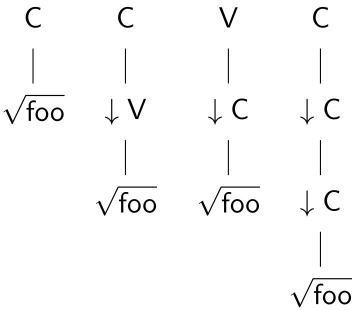
- What happens if every sentence must be a CP?

The crazy counting language

Every sentence must contain an odd number of *foo*-nodes.

- With **n** category features, grammar can count
  - up to **n**,
  - modulo **n**.
- We want neither!

Subcategorization overgenerates with only 2 features



- What happens if every sentence must be a CP?

The crazy counting language

Every sentence must contain an odd number of *foo*-nodes.

- With **n** category features, grammar can count
  - up to **n**,
  - modulo **n**.
- We want neither!

The danger of features

- Features can do much more than intended.
- Every feature opens up a new backdoor.

Features are redundant

- Features can be completely **replaced by constraints**.

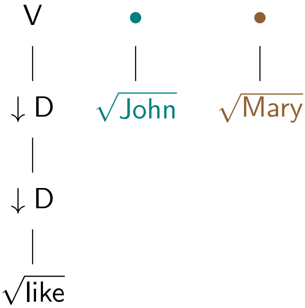
V — $\downarrow D$ — $\downarrow D$ — $\sqrt{\text{like}}$

D — $\sqrt{\text{John}}$

D — $\sqrt{\text{Mary}}$

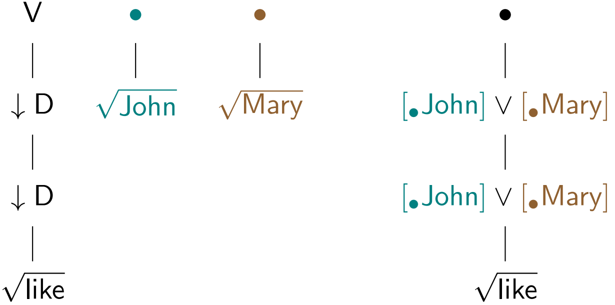
Features are redundant

- Features can be completely **replaced by constraints**.



Features are redundant

- Features can be completely **replaced by constraints**.



Features are redundant [cont.]

- Worst-case scenario: infinitary first-order logic

$$
\begin{array}{c}
\mathrm{D} \\ | \\ \downarrow \mathrm{D} \\ | \\ \downarrow \mathrm{N} \\ | \\ \sqrt{\text{'s}}
\end{array}
\qquad
\begin{array}{c}
\mathrm{D} \\ | \\ \sqrt{\text{John}}
\end{array}
\qquad
\begin{array}{c}
\mathrm{N} \\ | \\ \sqrt{\text{mother}}
\end{array}
\qquad
\begin{array}{c}
\bullet \\ | \\ [_\bullet \text{John}] \vee [_\bullet [_\bullet \text{John}][_\bullet \text{'s}[_\bullet \text{mother}]] \vee \cdots \\ | \\ [_\bullet \text{mother}] \\ | \\ \sqrt{\text{'s}}
\end{array}
$$

- But: the infinite disjunctions form **recognizable sets**
  ⇒ finite description via **monadic second order logic (MSO)**

Features $\equiv$ constraints

Interdefinability theorem (Graf 2011, 2013, 2017; Kobele 2011)

- Features can be replaced by MSO-definable constraints.
  - Strategy from previous slide
- Every MSO-definable constraint can be encoded via features.
  1. Represent constraint as machine with finitely many states.
  2. Category features also encode state of machine.

- One can also co-opt movement features, ϕ-features, ...
- Subcategorization is sufficient, though
  $\Rightarrow$ interdefinability holds for pretty much every framework

So what is a feature?

- Features and constraints are **two sides of the same coin**.

  Features: distributed encoding of constraint

  Constraints: global behavior arising from feature interactions

- Problem: the coin is too big...

Crazy MSO-definable constraints

- **modulo counting** (already seen)
- **symmetric opposites**
  NPI must c-command its licensor
- **Boolean constraint conjunction**
  satisfy either NPI-licensing or V2 iff Principle B is satisfied
- **number sensitivity**
  Principle A holds only if there's ≥ 3 reflexives
- **no locality**
  last word of first TP $=$ first word of last TP
- **domain mixing**
  if the first word is downward entailing,
  then the last word must not contain an onset cluster

Limiting overgeneration

- This massive overgeneration must be curtailed.
- Constraints are much easier to regulate than feature systems.

Methodological argument

- Features may well be real, but they are hard to rein in.
- Adopt constraint-based perspectives wherever possible.

Limiting overgeneration

- This massive overgeneration must be curtailed.
- Constraints are much easier to regulate than feature systems.

Methodological argument

- Features may well be real, but they are hard to rein in.
- Adopt constraint-based perspectives wherever possible.

Linguist "Hold on a sec, I know how to fix features. . . "

Me "You might, but so far we have no working solution."

Banning new features

- Stipulate a fixed, universal set of features

Problems

1. This will be a very large set.
   - Treebanks have hundreds of features for just one language.
   - We need dozens of formal features for various movement steps.
2. Any sufficiently large set will allow for crazy constraints.
   - Remember, 2 features already give us modulo counting.
3. Formal universals are preferable to substantive universals.

Respect feature content

- Features may only be used according to their content (e.g. encoding number)

Problems

1. Sounds good, but how do you enforce it?
2. How does on rule out every conceivable kind of feature abuse?
3. Many features have no content (e.g. movement).
4. What is the content of a category feature?
5. Begs the question: what is a feature, what is a constraint?

Respect feature content

- Features may only be used according to their content (e.g. encoding number)

Problems

1. Sounds good, but how do you enforce it?
2. How does on rule out every conceivable kind of feature abuse?
3. Many features have no content (e.g. movement).
4. What is the content of a category feature?
5. Begs the question: what is a feature, what is a constraint?

“I know it when I see it.”

Supreme Court Justice Potter Steward

Feature algebras

- Set of features = algebraic structure with specific entailments (Harbour 2015)

Example

- If you can select N, then you can also select Num.

Problems

1. If subcategorization cannot be fixed, then fixing other parts of the feature calculus is pointless.
2. There seems to be no obvious category algebra, thus leaving subcategorization unconstrained.
3. Empirical exploration is hard because the notion of category is very fuzzy.

Feature independence

- Features come in blocks (category, selector, ϕ, ...)
- Choice of features in one block is independent of other blocks

Problems

1. Empirical issues
   - $\sqrt{\text{water}}$ carries $\downarrow$D iff it carries V
   - mass/count features only available with N
2. If category must be in same block as subcategorization or subtype features, we're back to square 1.

Drop subcategorization/c-selection

- Merge is free and there is only s-selection at the interfaces.

Problems

1. Admitting defeat, putting all the work into constraints
2. There's still ϕ-features, movement features, etc., and those can be abused too.

Why features are hard to rein in

- Features produce global behavior through small interactions.
- The behavior is encoded in a distributed fashion over thousands of roots.
- It is very hard to relate the high-level behavior to specific aspects of the feature calculus.

Analogies

- Writing in assembly code
- Using quantum mechanics to model falling leaves

- Constraints are much easier to restrict.

Constraints below MSO

- We can limit constraints to specific complexity classes.
- This is all the rage now in **subregular phonology**.

Class	**Relevance**
FO	safe upper bound for phonotactics
IBSP	constraints limited to unbounded locality domain
TSL	relativized locality
SL	strict locality

Jeff Heinz

Jane Chandlee

Adam Jardine

Kevin McMullin

Extension to trees

- More recently, there's similar work on **subregular syntax**.

Class	Relevance
FO	super-safe upper bound for syntax (Graf 2012)
STA	potential upper bound on syntax (Graf and De Santo 2019)
IBSP	island constraints (Shafie and Graf 2019)
TSL	c-command licensing (Graf and Shafiei 2019; Vu 2018)
	case assignment(?) (Vu et al. 2019)
SL	selection

Aniello De Santo

Sabine Laszakovits

Nazila Shafiei

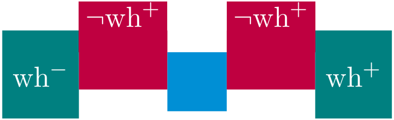
Mai Ha Vu

Example: IBSP constraint in phonology

- Tone plateauing: no L between two H in same word
- LHHHHHL, LHLLLLL, *LHLLLHL, LHL$LHL

Example: IBSP analysis of Complex NP island

- *What did you hear [$_{\text{NP}}$ rumors [$_{\text{CP}}$ that John bought _]]?
- **Selection sequence/Ancestor chain**:
$\sqrt{\text{what}}[\mathrm{wh}^-] \prec \sqrt{\text{buy}} \prec \text{T} \prec \sqrt{\text{that}} \prec \sqrt{\text{rumor}}[\downarrow \text{C}, \text{N}] \prec \sqrt{\text{hear}} \prec \sqrt{\text{do}} \prec \text{C}[\mathrm{wh}^+]$



$\Rightarrow$ *[$\downarrow$ C, N]

- (Most) island constraints are syntactic counterpart to circumambient patterns in phonology.
- If the constraint had been expressed purely via features, this parallel would have been lost.

Feature remnants

- All this work still uses features to encode essential information.
  e.g. what moves where
- It also looks mostly at phenomena that are commonly analyzed with constraints.
- It's harder to **remove features** from feature-based analyses.
- This is largely unexplored territory.
  even GB had indices as meta-features for movement
- But it's worth the effort, imho.

Some case studies

1. Features ≡ constraints
   - Even 2 features is too much
   - All features are redundant
2. Limiting expressivity
   - Via features: Failure
   - Via constraints: Success
3. Case studies
   - Successive cyclic movement
   - Categories and selection
   - Morphosyntax

Successive cyclic movement without features

- Successive cyclic movement does not require movement features because landing sites are predictable.
- What did you hear [$_{\text{CP}}$ that John bought _]?
- **Selection sequence/Ancestor chain**: $\sqrt{\text{what}}[\mathrm{wh}^-] \prec \sqrt{\text{buy}} \prec \text{T} \prec \sqrt{\text{that}}[\text{C},\mathrm{wh}^+] \prec \sqrt{\text{hear}} \prec \sqrt{\text{do}} \prec \text{C}[\mathrm{wh}^+]$
- C-head doesn't need wh^+; it's a landing site by virtue of occurring along the movement path.
- If some computation moves *what* to Spec,CP due to wh^+, then it can also do so because of C.

Or perhaps I'm wrong?

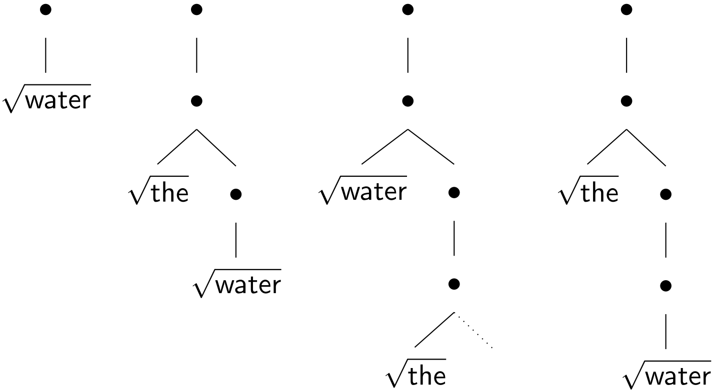
General upshot

- Movement only needs features on targets that the computational machinery cannot detect otherwise.

- Looking forward to Michelle's talk.
- Will there be a way to state the feature distinction purely in terms of constraints?
- Also: my account **still relies on category features**!

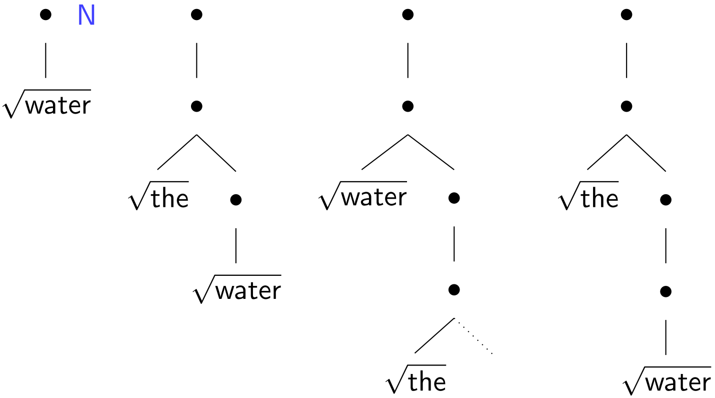
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



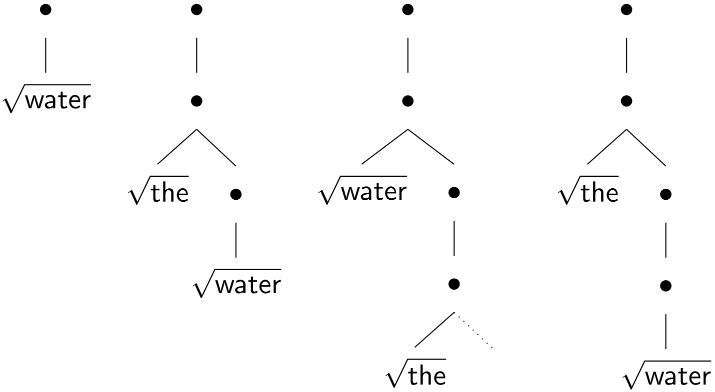
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



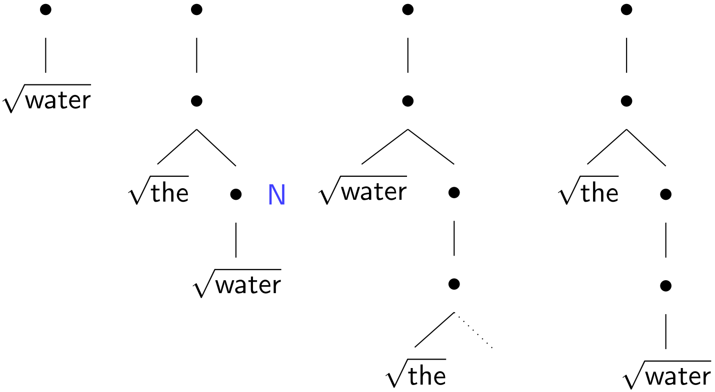
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



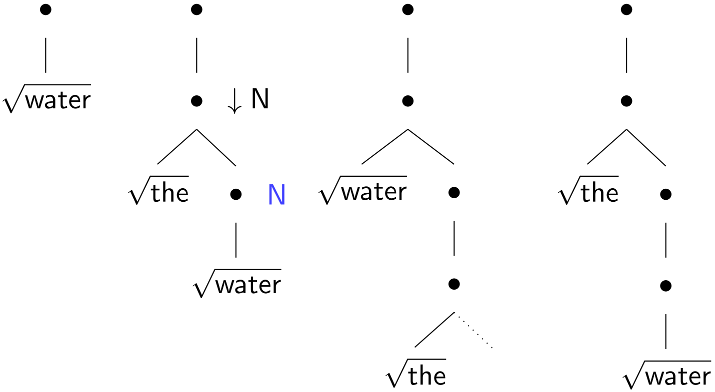
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



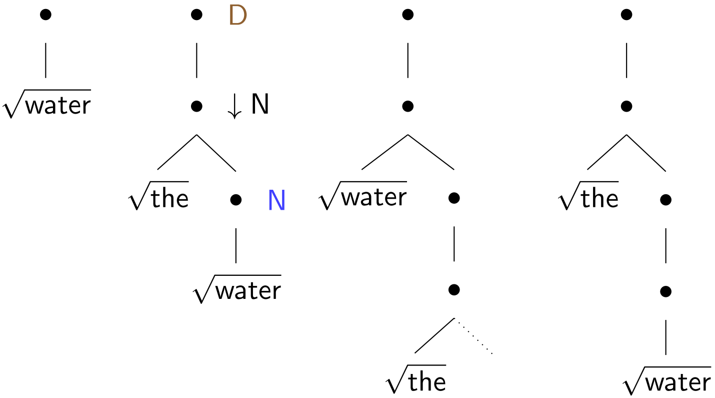
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



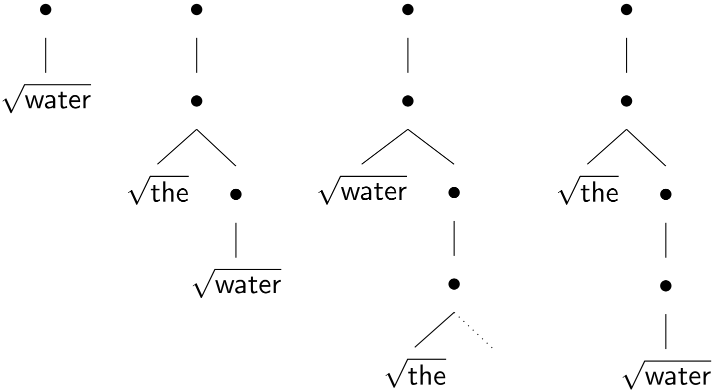
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



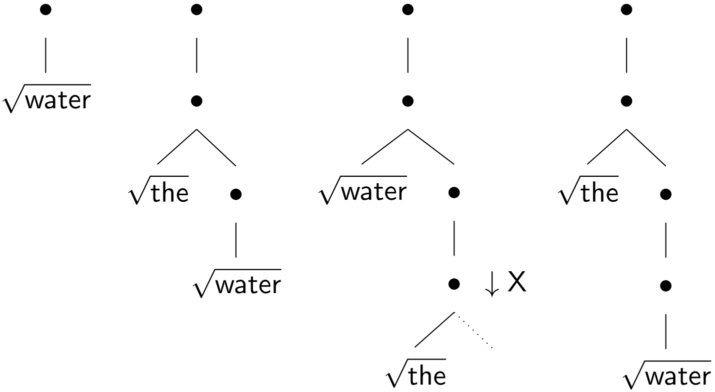
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



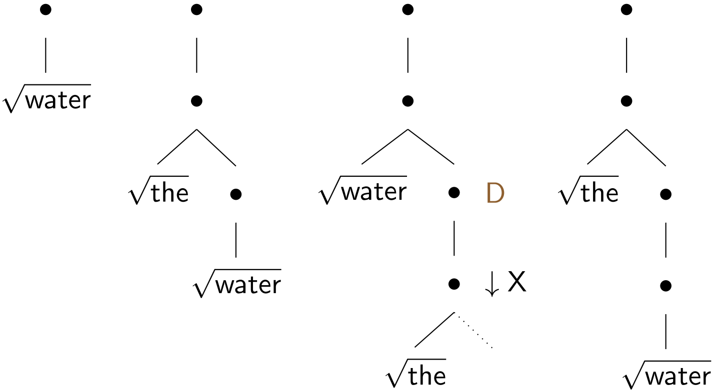
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



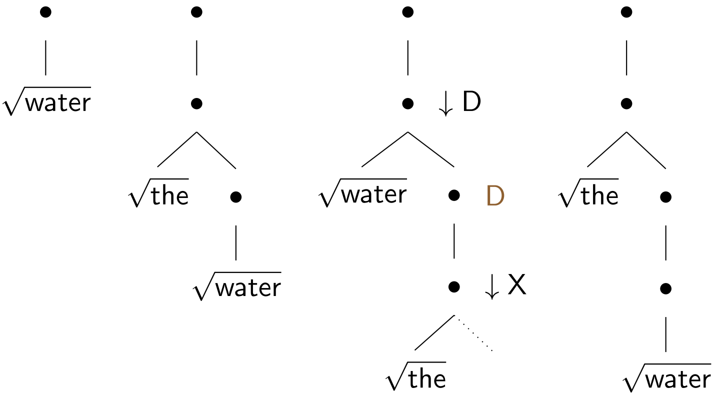
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



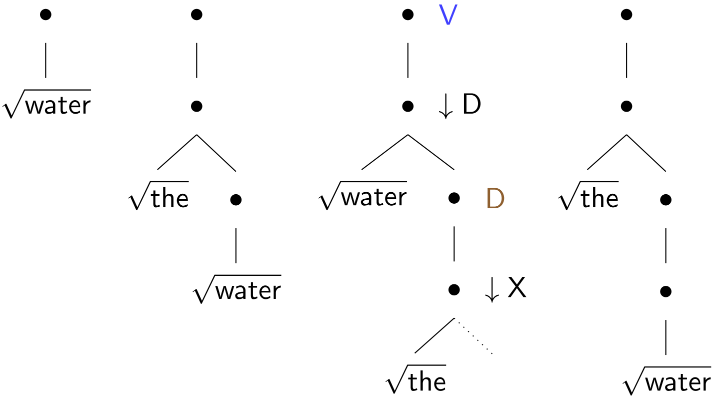
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



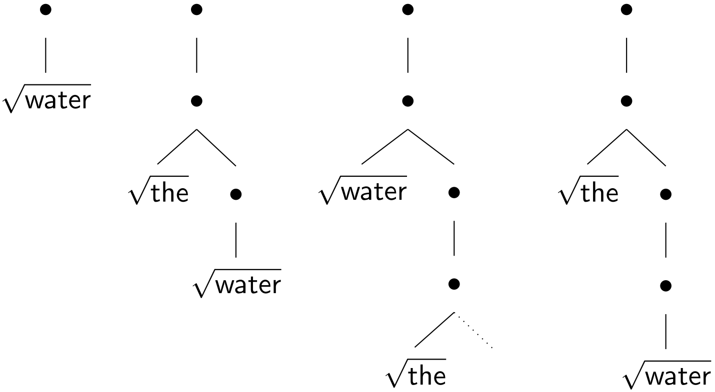
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



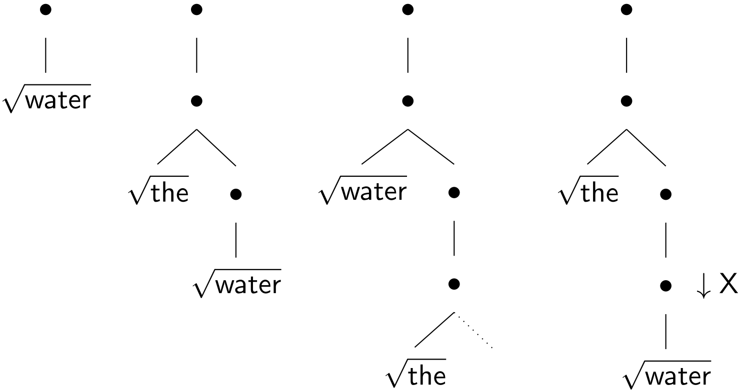
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



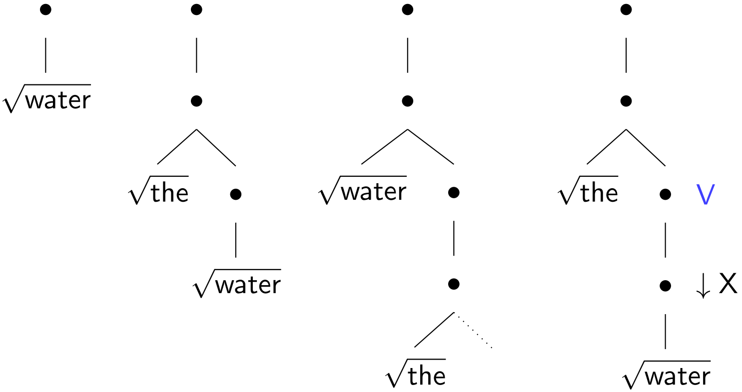
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



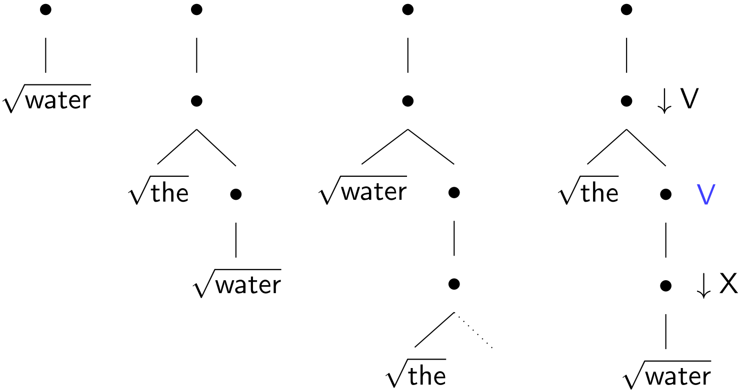
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



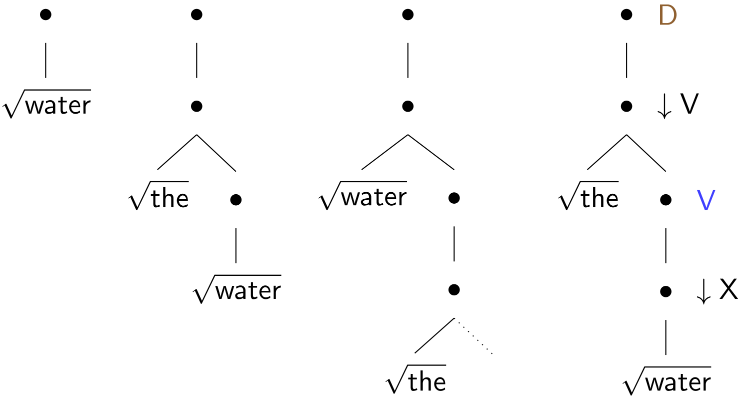
Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



Doing away with category and selector features

- Suppose a root is never explicitly assigned a category.
- Instead, its category is inferred from the **local context**.



An empirical prediction

Locally bounded category ambiguity

Any root that is categorially ambiguous must be disambiguated by its local context.

Example: Unbounded categorial dependency

- Suppose we can have the following two ancestor chains:

$$\begin{array}{ccccccccc} A[X] & \prec & B_1[X_1] & \prec & \cdots & \prec & B_n[X_n] & \prec & C[X] \\ A[Y] & \prec & B_1[Y_1] & \prec & \cdots & \prec & B_n[Y_n] & \prec & C[Y] \end{array}$$

- The category feature of C ultimately depends on the category feature of A.
- This is possible with category features, but not local inference.

Linguistic use of ditching features

- Unbounded categorial dependencies don't arise in syntax.
- Category features incorrectly allow for them,
  whereas the feature-free account does not.
- The restriction to local contexts puts selection in the class SL,
  the weakest known class of constraints.

General upshot

- Doing away with features isn't just a formal enterprise.
- It also furnishes new empirical generalizations and predictions.

Feature-free morphosyntax: *ABA

*ABA generalization (Bobaljik 2012)

Given an underlying hierarchy **x** $>$ **y** $>$ **z**,
z cannot pattern with **x** to the exclusion of **y**.

Example

- Adjectival gradation: *good — better — goodest
- Pronoun paradigm: *I — you — I (cf. Harbour 2015)

Usually explained via

- layered feature hierarchies (Bobaljik 2012)
- feature combinatorics (Bobaljik and Sauerland 2017)

*ABA with algebras

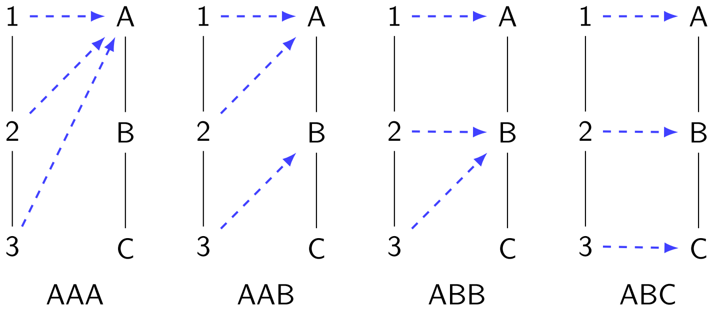
- We can give a feature-free account via algebras.

Monotonicity

- Monotonicity is a property of functions.
- For our purposes:
  If $\mathbf{x} < \mathbf{y} < \mathbf{z}$, then it cannot hold that $f(\mathbf{x}) = f(\mathbf{z}) \neq f(\mathbf{y})$.

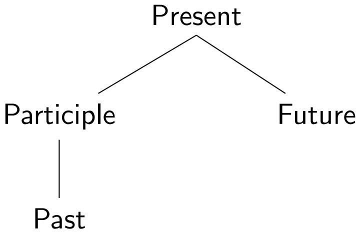
- Monotonicity immediately derives *ABA
  for any linear order with 3 elements.

*ABA for adjectives and pronouns



	1	**2**	**3**
adjectives	positive	comparative	superlative
pronouns	1p form	2p form	3p form

Beyond 3 elements: *ABA for tense



- **Forbids**:
  present = past ≠ participle
- **Allows**:
  present = participle ≠ future, past
  present = future ≠ participle = past
  present ≠ participle = future ≠ past

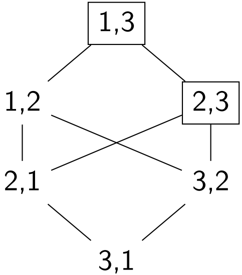
Sophie Moradi

*ABA for PCC

- Monotonicity even handles the attested PCCs. (Graf 2019)

IO↓/DO→	1	2	3
1	NA	*	✓
2	*	NA	✓
3	*	*	NA

S-PCC

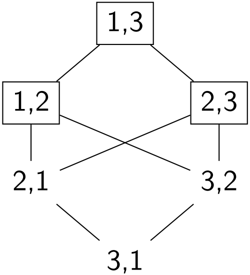


*ABA for PCC

- Monotonicity even handles the attested PCCs. (Graf 2019)

IO$\downarrow$/DO$\rightarrow$	1	2	3
1	NA	✓	✓
2	*	NA	✓
3	*	*	NA

U-PCC

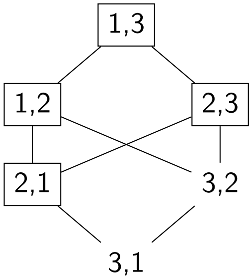


*ABA for PCC

- Monotonicity even handles the attested PCCs. (Graf 2019)

IO↓/DO→	1	2	3
1	NA	✓	✓
2	✓	NA	✓
3	*	*	NA

W-PCC

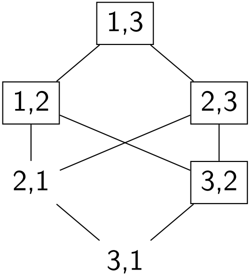


*ABA for PCC

- Monotonicity even handles the attested PCCs. (Graf 2019)

IO↓/DO→	1	2	3
1	NA	✓	✓
2	*	NA	✓
3	*	✓	NA

M-PCC



Total empirical coverage of monotonicity + algebras

- Adjectival gradation
- Pronoun syncretism
- Tense
- Case syncretism
- Noun stem allomorphy
- PCC
- Gender-Case Constraint
- and extends far beyond morphosyntax
  - No Crossing Branches constraint
  - Ban against improper movement
  - Williams cycle
  - NPI licensing
  - Keenan-Comrie hierarchy
  - 3/4 splits (e.g. in expletive negation)
  - and more

Nature of the hierarchies

- The hierarchies arrange realizational classes, not features!
  e.g. 3rd person might not be a feature, but it's a realizational class
- Even where features may be involved,
  they may look very different across domains.
  person in morphology $\neq$ person in PCC

General upshot

- Algebras offer a higher-level description of morphosyntax.
- Montonicity as a uniform constraint across many domains that would look very different at feature level.

Nature of the hierarchies

- The hierarchies arrange realizational classes, not features!
  e.g. 3rd person might not be a feature, but it's a realizational class
- Even where features may be involved,
  they may look very different across domains.
  person in morphology $\neq$ person in PCC

General upshot

- Algebras offer a higher-level description of morphosyntax.
- Montonicity as a uniform constraint across many domains that would look very different at feature level.

We'll see tomorrow how this gels with Omer's story.

Conclusion: My stance against features

- Features carry the risky of serious overgeneration.
- Constraints do so too, but we have better tools for studying (and limiting!) them.
- We can handwave this away as a mathematical curiosity but
  1. generative grammar values formal precision,
  2. that would be a lost opportunity.
- Focus on constraints over features furnishes new insights:
  - parallels to phonology (e.g. island constraints)
  - absence of unbounded categorial dependencies
  - cross-domain constraints in morphosyntax

What I hope to learn more about to(day/morrow)

- What is the motivation for feature-based accounts?
- Are there any cases where feature-based accounts are more insightful than constraint-based ones?
- What is a feature-theory a theory of?

Amendment to the take-home message

- I'm not against features on principled/conceptual grounds.
- If this workshop reveals new ways of restricting features, it's been worth the trip.

What I hope to learn more about to(day/morrow)

- What is the motivation for feature-based accounts?
- Are there any cases where feature-based accounts are more insightful than constraint-based ones?
- What is a feature-theory a theory of?

Amendment to the take-home message

- I'm not against features on principled/conceptual grounds.
- If this workshop reveals new ways of restricting features, it's been worth the trip.

References I

Bobaljik, Jonathan D. 2012. *Universals in comparative morphology: Suppletion, superlatives, and the structure of words*. Cambridge, MA: MIT Press.

Bobaljik, Jonathan D., and Uli Sauerland. 2017. *ABA and the combinatorics of morphological features. URL http://ling.auf.net/lingbuzz/003320, ms., University of Connecticut and Leibniz-Zentrum für Allgemeine Sprachwissenschaft (ZAS).

Graf, Thomas. 2011. Closure properties of Minimalist derivation tree languages. In *LACL 2011*, ed. Sylvain Pogodalla and Jean-Philippe Prost, volume 6736 of *Lecture Notes in Artificial Intelligence*, 96–111. Heidelberg: Springer. URL https://dx.doi.org/10.1007/978-3-642-22221-4_7.

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

Graf, Thomas. 2013. *Local and transderivational constraints in syntax and semantics*. Doctoral Dissertation, UCLA. URL http://thomasgraf.net/doc/papers/Graf13Thesis.pdf.

Graf, Thomas. 2017. A computational guide to the dichotomy of features and constraints. *Glossa* 2:1–36. URL https://dx.doi.org/10.5334/gjgl.212.

References II

Graf, Thomas. 2019. Monotonicity as an effective theory of morphosyntactic variation. *Journal of Language Modelling* To appear.

Graf, Thomas, and Aniello De Santo. 2019. Sensing tree automata as a model of syntactic dependencies. Ms., Stony Brook University.

Graf, Thomas, and Nazila Shafiei. 2019. C-command dependencies as TSL string constraints. In *Proccedings of SCiL 2019*, ed. Gaja Jarosz, Max Nelson, Brendan O'Connor, and Joe Pater, 205–215.

Harbour, Daniel. 2015. Poor pronoun systems and what they teach us. *Nordlyd* 41. URL `http://dx.doi.org/10.7557/12.3314`.

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. URL `https://doi.org/10.1007/978-3-642-22221-4_9`.

Shafie, Nazila, and Thomas Graf. 2019. The subregular complexity of syntactic islands. Ms., Stony Brook University.

Vu, Mai Ha. 2018. Towards a formal description of NPI-licensing patterns. In *Proceedings of the Society for Computation in Linguistics*, volume 1, 154–163.

Vu, Mai Ha, Nazila Shafiei, and Thomas Graf. 2019. Case assignment in TSL syntax: A case study. In *Proccedings of SCiL 2019*, ed. Gaja Jarosz, Max Nelson, Brendan O'Connor, and Joe Pater, 267–276.