Ling 566
Oct 26, 2023

Lexical Types

Overview

- Motivation for lexical hierarchy
- Default inheritance
- Tour of the lexeme hierarchy
- The Case Constraint
- *pos* vs. *lexeme*
- Reading Questions

Motivation

- We've streamlined our grammar rules...
- ...by stating some constraints as general principles
- ...and locating lots of information in the lexicon.
- Our lexical entries currently stipulate a lot of information that is common across many entries and should be stated only once.
- Examples?
- Ideally, particular lexical entries need only give phonological form, the semantic contribution, and any constraints truly idiosyncratic to the lexical entry.

Lexemes and Words

- **Lexeme**: An abstract proto-word which gives rise to genuine words. We refer to lexemes by their 'dictionary form', e.g. 'the lexeme *run*' or 'the lexeme *dog*'.
- **Word:** A particular pairing of form and meaning. *Running* and *ran* are different words

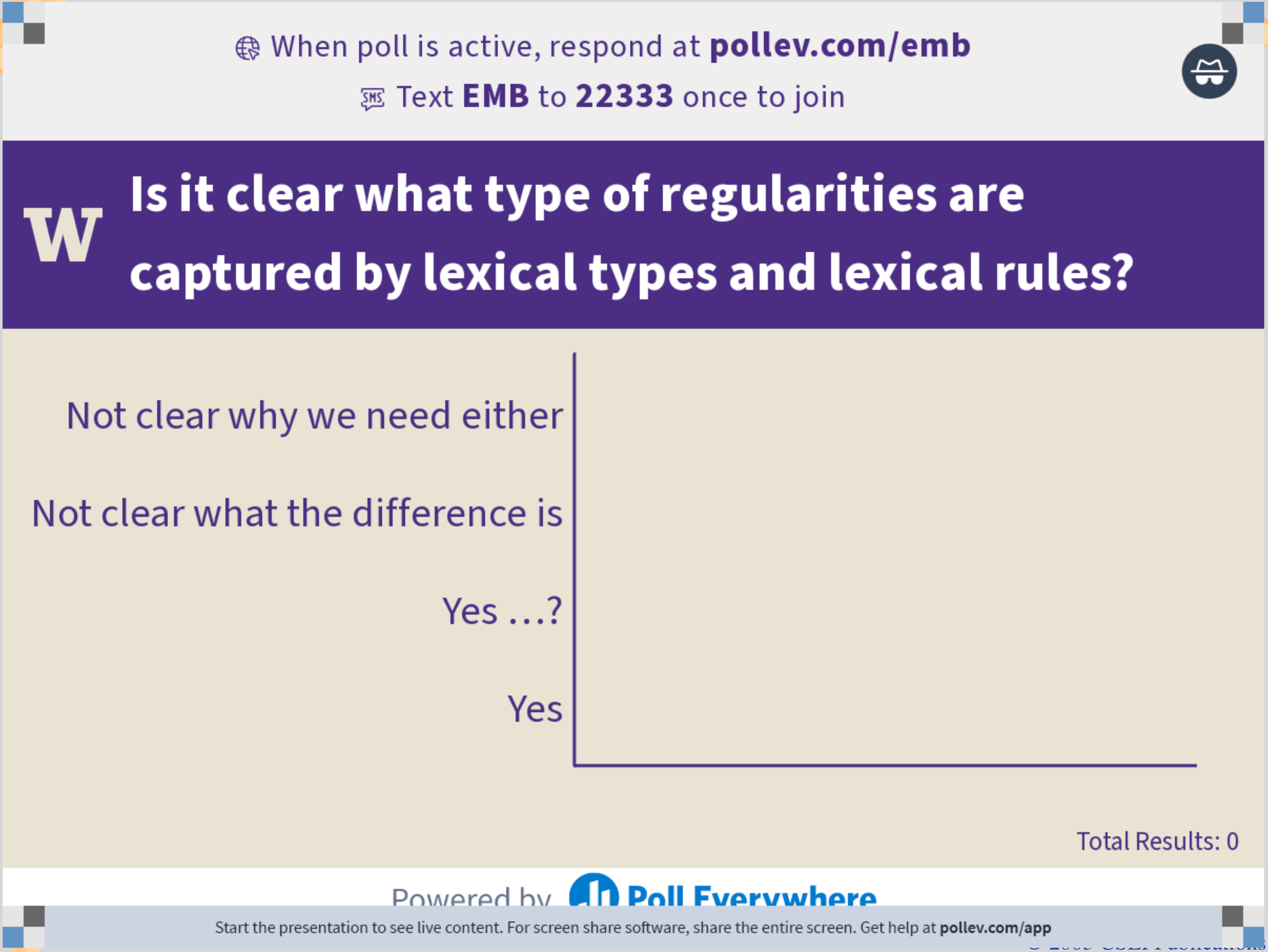
Q: Is lexeme the same as lemma?

Lexical Types & Lexical Rules

- Lexemes capture the similarities among *run, runs, running,* and *run*.
- The lexical type hierarchy captures the similarities among *run, sleep,* and *laugh*, among those and other verbs like *devour* and *hand*, and among those and other words like *book*.

  Q: What do *devour* and *book* have in common?

  A: The SHAC
- Lexical rules capture the similarities among *runs*, *sleeps, devours, hands*,...



When poll is active, respond at **pollev.com/emb**

Text **EMB** to **22333** once to join

Is it clear what type of regularities are captured by lexical types and lexical rules?

Not clear why we need either

Not clear what the difference is

Yes …?

Yes

Total Results: 0

Powered by Poll Everywhere

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

Default Inheritance

Q: Why do we have default inheritance?

A: Generalizations with exceptions are common:

- Most nouns in English aren't marked for CASE, but pronouns are.
- Most verbs in English only distinguish two agreement categories (*3sing* and *non-3sing*), but *be* distinguishes more.
- Most prepositions in English are transitive, but *here* and *there* are intransitive.
- Most nominal words in English are 3rd person, but some (all of them pronouns) are 1st or 2nd person.
- Most proper nouns in English are singular, but some (mountain range names, sports team names) are plural.

Default Inheritance, Technicalities

If a type says
ARG-ST / < NP >,
and one of its
subtypes says
ARG-ST < >,
then the ARG-ST
value of instances of
the subtype is < >.

If a type says
ARG-ST < NP >,
and one of its
subtypes says
ARG-ST < >,
then this subtype can
have no instances,
since they would
have to satisfy
contradictory
constraints.

Default Inheritance, More Technicalities

- If a type says MOD / < S >, and one of its subtypes says MOD <[SPR < NP>] >, then the MOD value of instances of the subtype is what?

$$\begin{bmatrix} \text{MOD} & \left\langle \begin{bmatrix} \text{HEAD} & / \; \textit{verb} \\ \text{SPR} & \langle \text{NP} \rangle \\ \text{COMPS} & / \; \langle \; \rangle \end{bmatrix} \right\rangle \end{bmatrix}$$

- That is, default constraints are ‘pushed down’

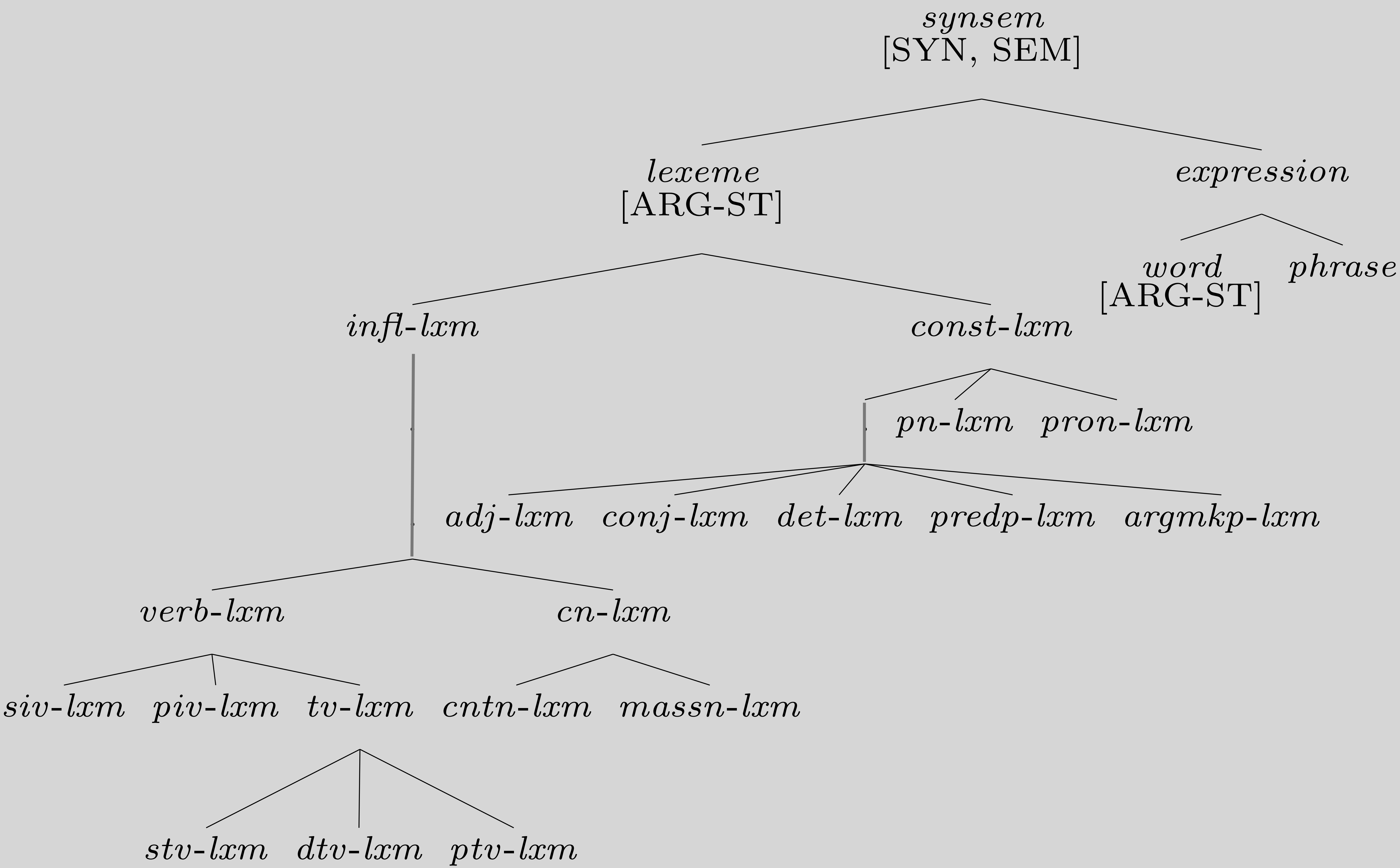
Question on Default Inheritance

Q: Can a grammar rule override a default constraint on a word?

A: No. Defaults are all ‘cached out’ in the lexicon.

- Words as used to build sentences have only inviolable constraints.

Our Lexeme Hierarchy



synsem
[SYN, SEM]

lexeme
[ARG-ST]

expression

word
[ARG-ST]

phrase

infl-lxm

const-lxm

pn-lxm

pron-lxm

adj-lxm

conj-lxm

det-lxm

predp-lxm

argmkp-lxm

verb-lxm

cn-lxm

siv-lxm

piv-lxm

tv-lxm

cntn-lxm

massn-lxm

stv-lxm

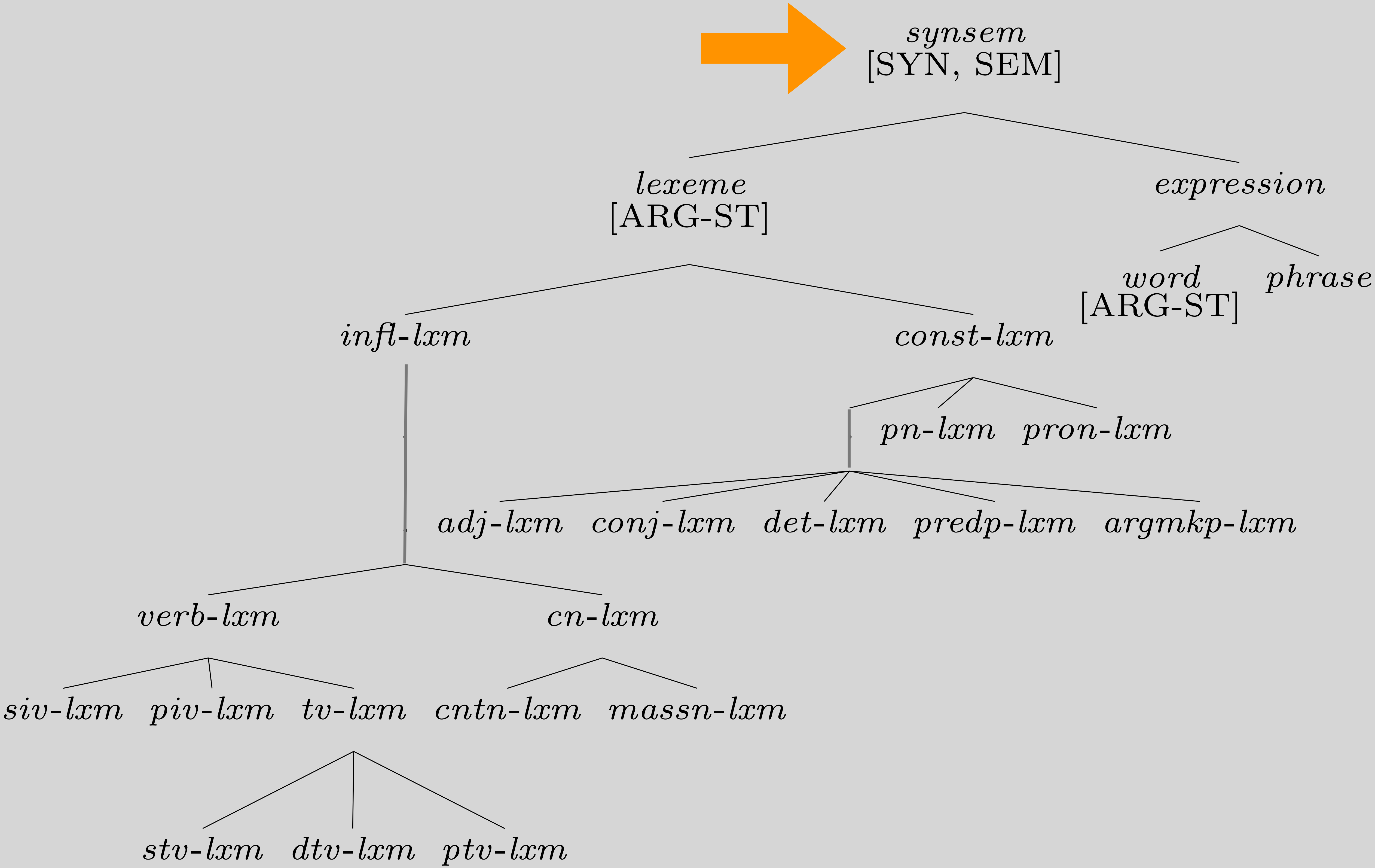
dtv-lxm

ptv-lxm

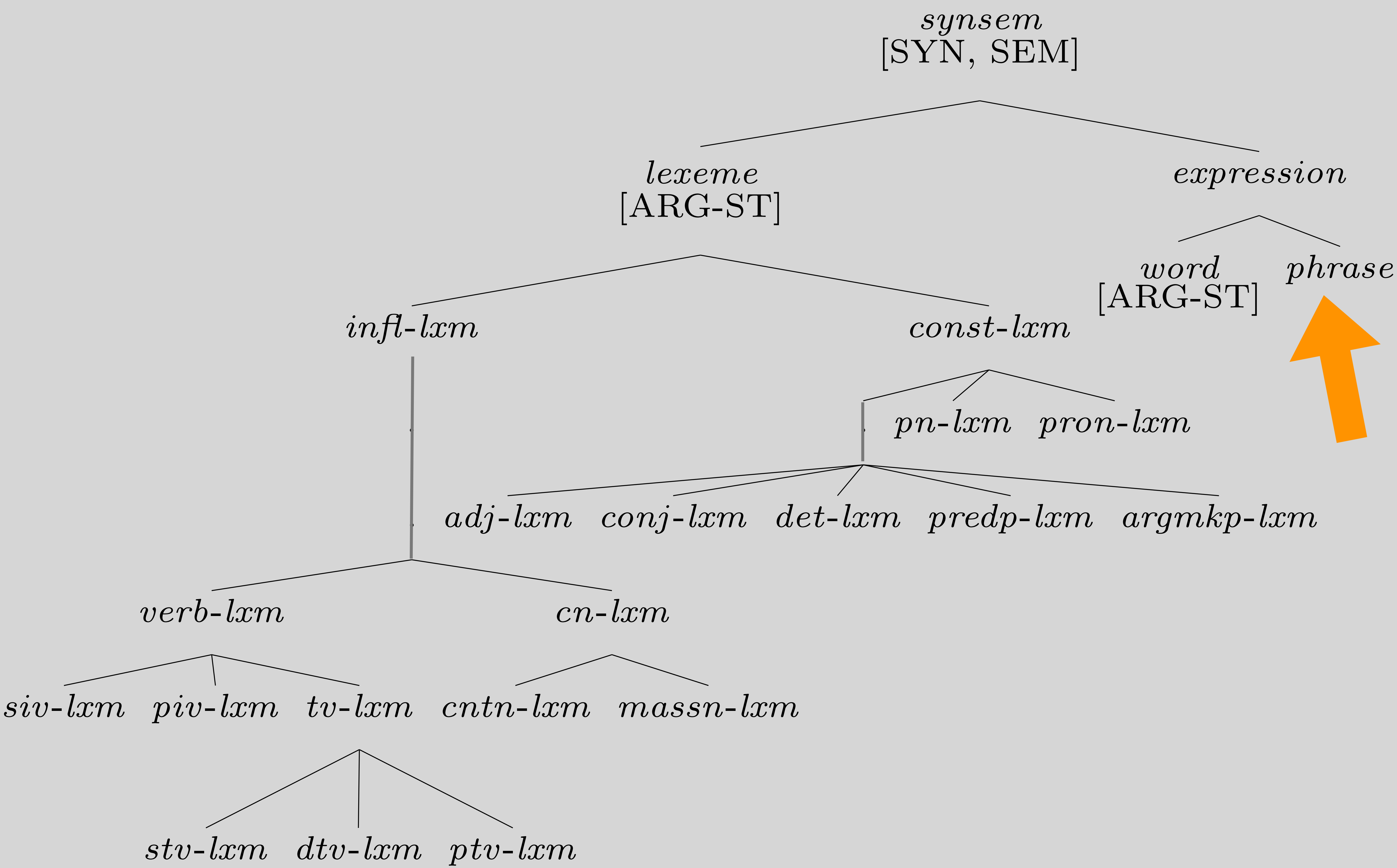
Functions of Types

- Stating what features are appropriate for what categories
- Stating generalizations
- Constraints that apply to (almost) all instances
- Generalizations about selection -- where instances of that type can appear

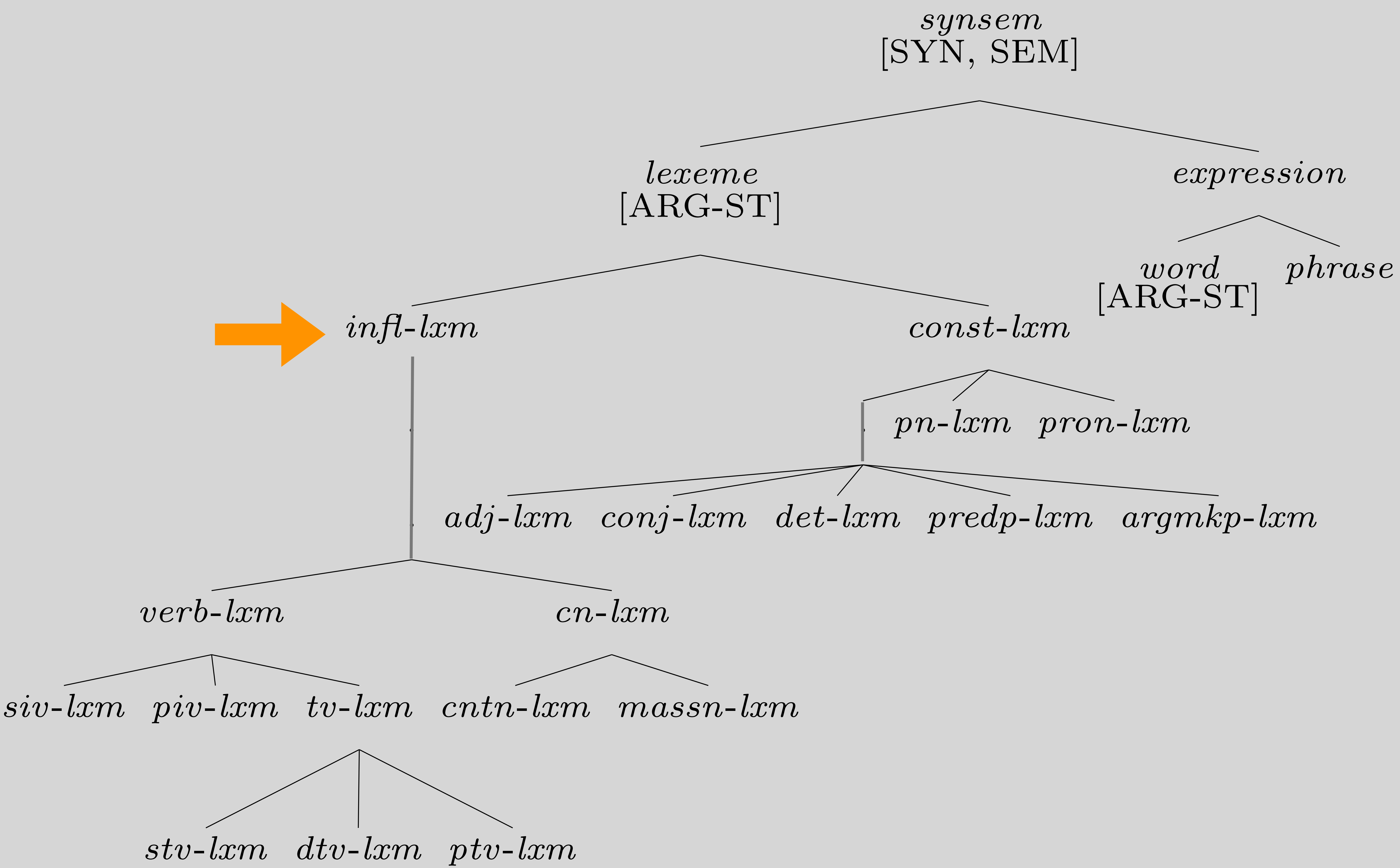
Every *synsem* has the features SYN and SEM



No ARG-ST on *phrase*



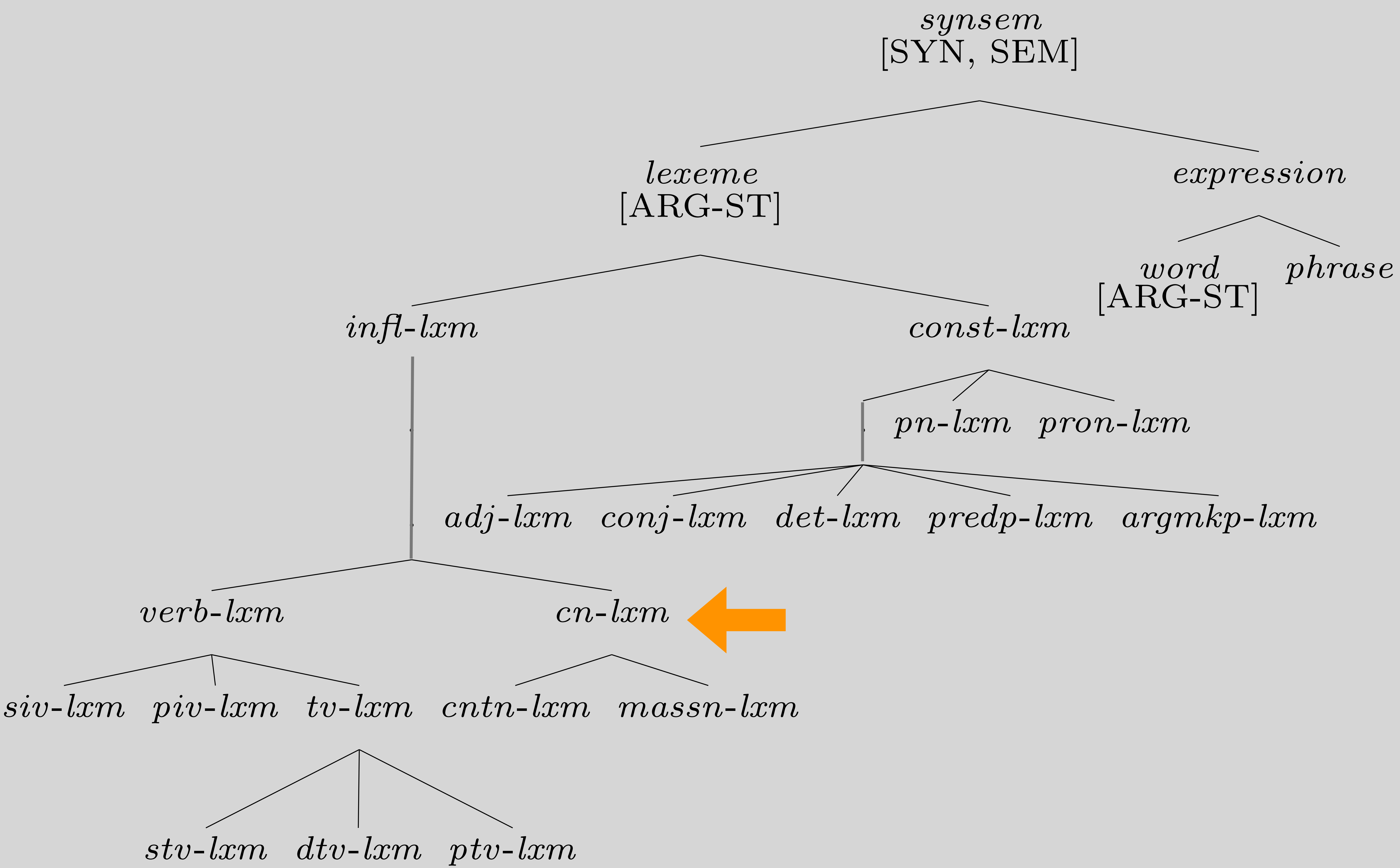
A Constraint on *infl-lxm*: the SHAC



A Constraint on *infl-lxm*: the SHAC

$$\textit{infl-lxm}: \left[\text{SYN} \left[\begin{array}{ll} \text{VAL} & \left[\text{SPR} \left\langle \left[\text{AGR} \quad \boxed{1}\right] \right\rangle\right] \\ \text{HEAD} & \left[\text{AGR} \quad \boxed{1}\right] \end{array}\right]\right]$$

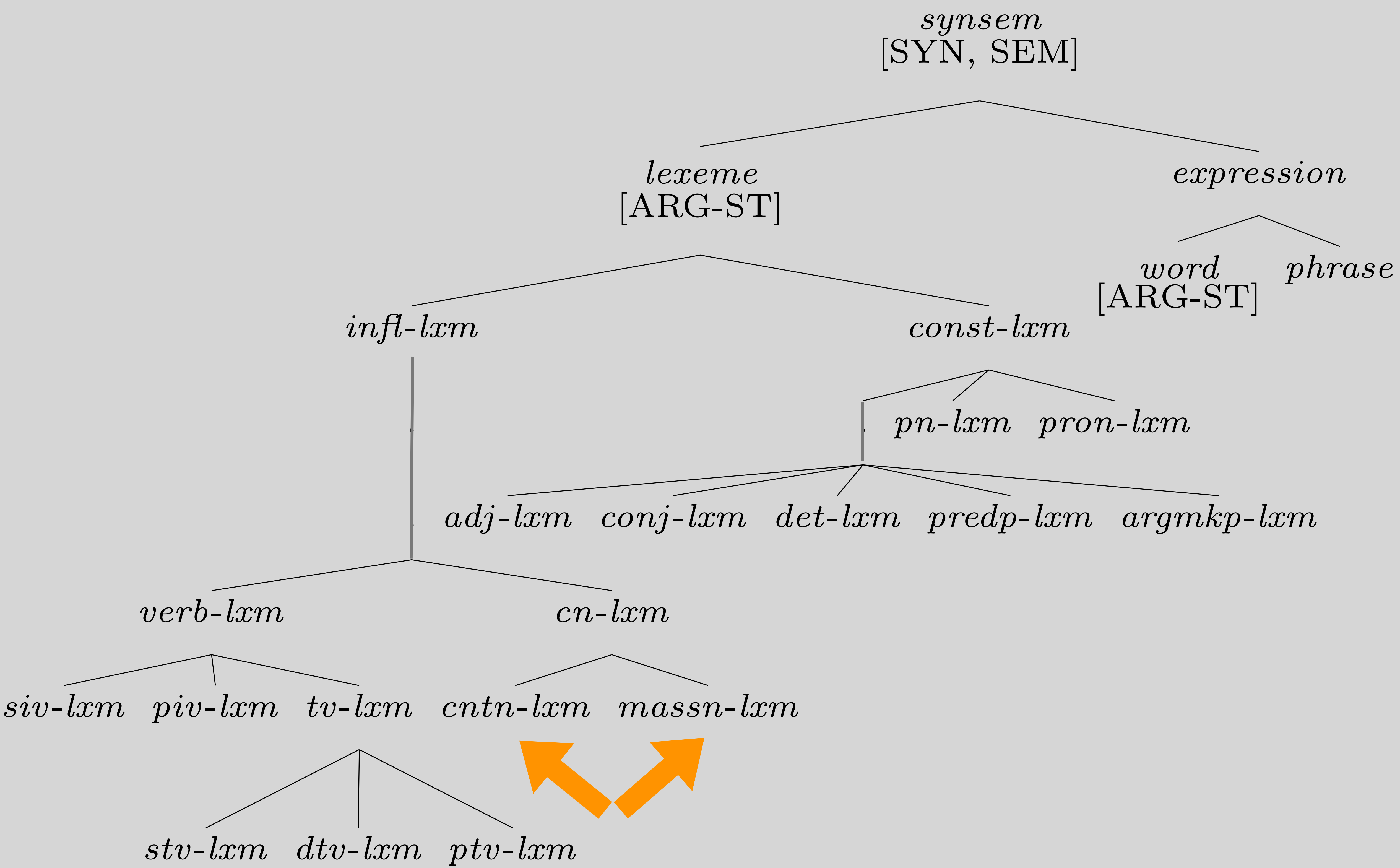
Constraints on *cn-lxm*



Constraints on *cn-lxm*

$$
\textit{cn-lxm}: \begin{bmatrix}
\text{SYN} & \begin{bmatrix}
\text{HEAD} & \begin{bmatrix} \textit{noun} & \\ \text{AGR} & [\text{PER 3rd}] \end{bmatrix} \\
\text{VAL} & \begin{bmatrix} \text{SPR} & \left\langle \begin{bmatrix} \text{HEAD} & \text{det} \\ \text{INDEX} & i \end{bmatrix} \right\rangle \end{bmatrix}
\end{bmatrix} \\
\text{SEM} & \begin{bmatrix} \text{MODE} & /\ \text{ref} \\ \text{INDEX} & i \end{bmatrix} \\
\text{ARG-ST} & \langle \text{X} \rangle \oplus /\langle\ \rangle
\end{bmatrix}
$$

Formally Distinguishing Count vs. Mass Nouns

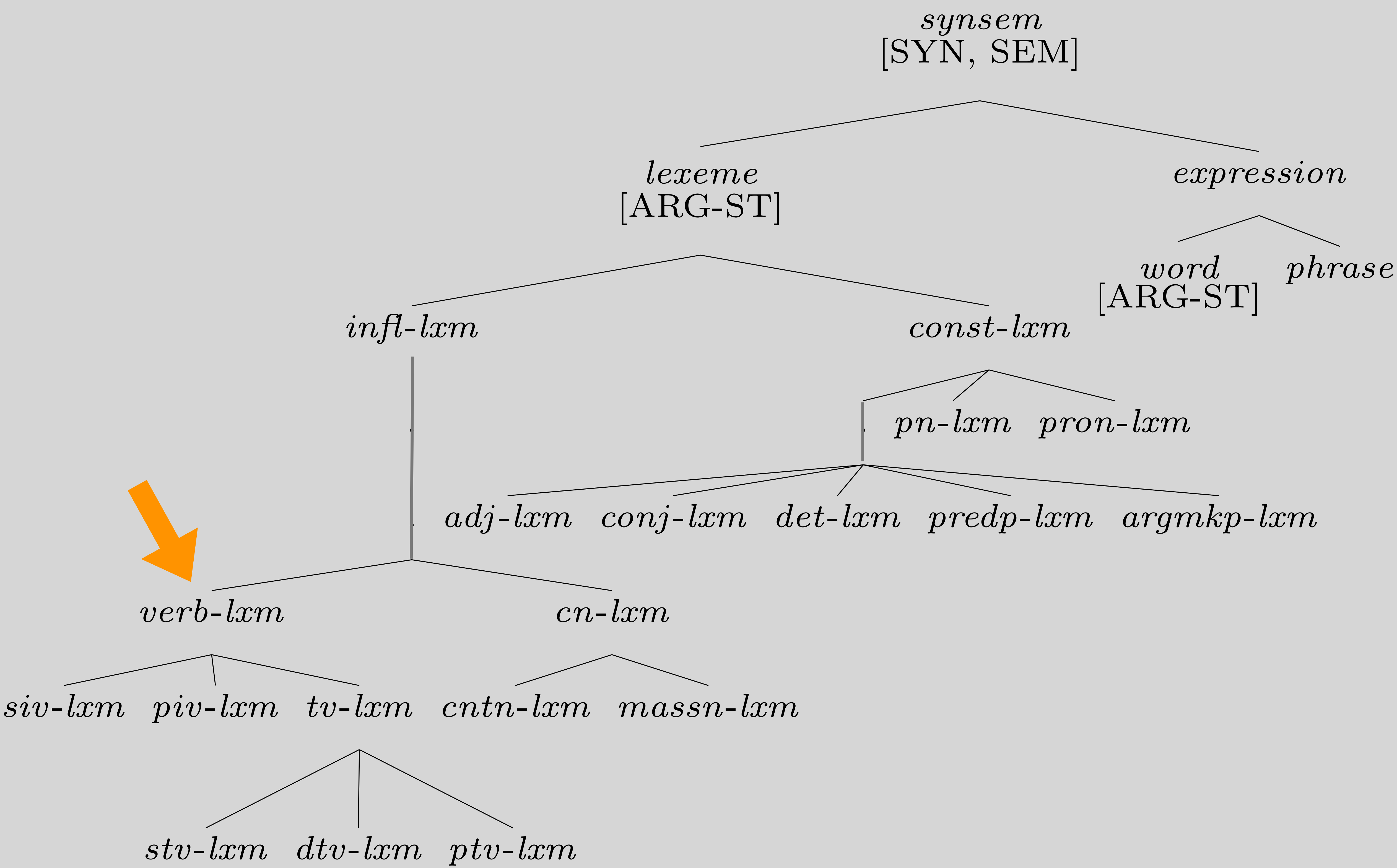


Formally Distinguishing Count vs. Mass Nouns

$$\textit{cntn-lxm} : \begin{bmatrix}\text{SYN} & \begin{bmatrix}\text{VAL} & \begin{bmatrix}\text{SPR} & \langle\ [\text{COUNT}\ +]\ \rangle\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

$$\textit{massn-lxm} : \begin{bmatrix}\text{SYN} & \begin{bmatrix}\text{VAL} & \begin{bmatrix}\text{SPR} & \langle\ [\text{COUNT}\ -]\ \rangle\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

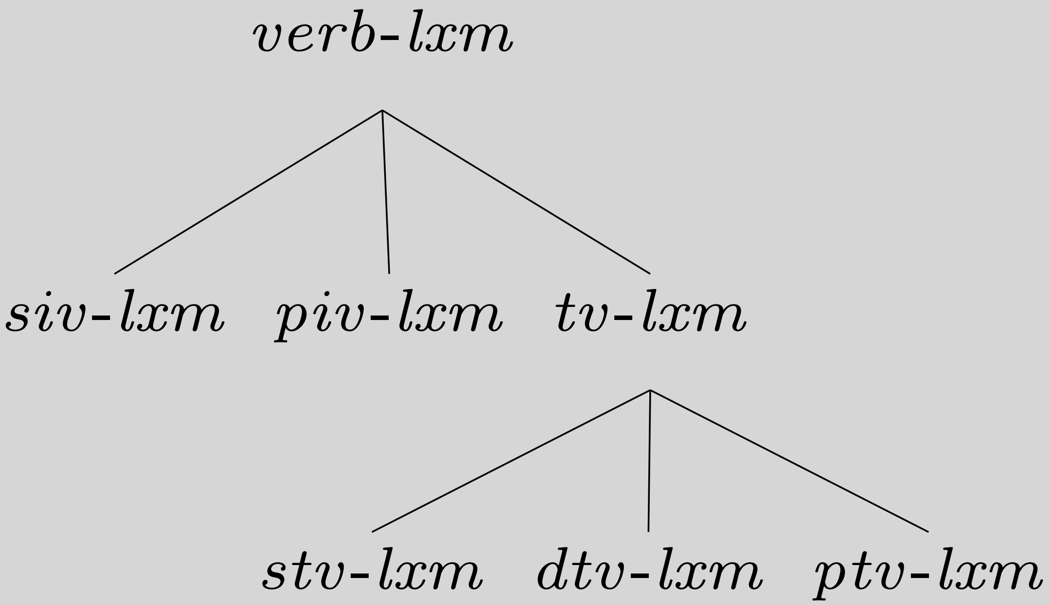
Constraints on *verb-lxm*



Constraints on *verb-lxm*

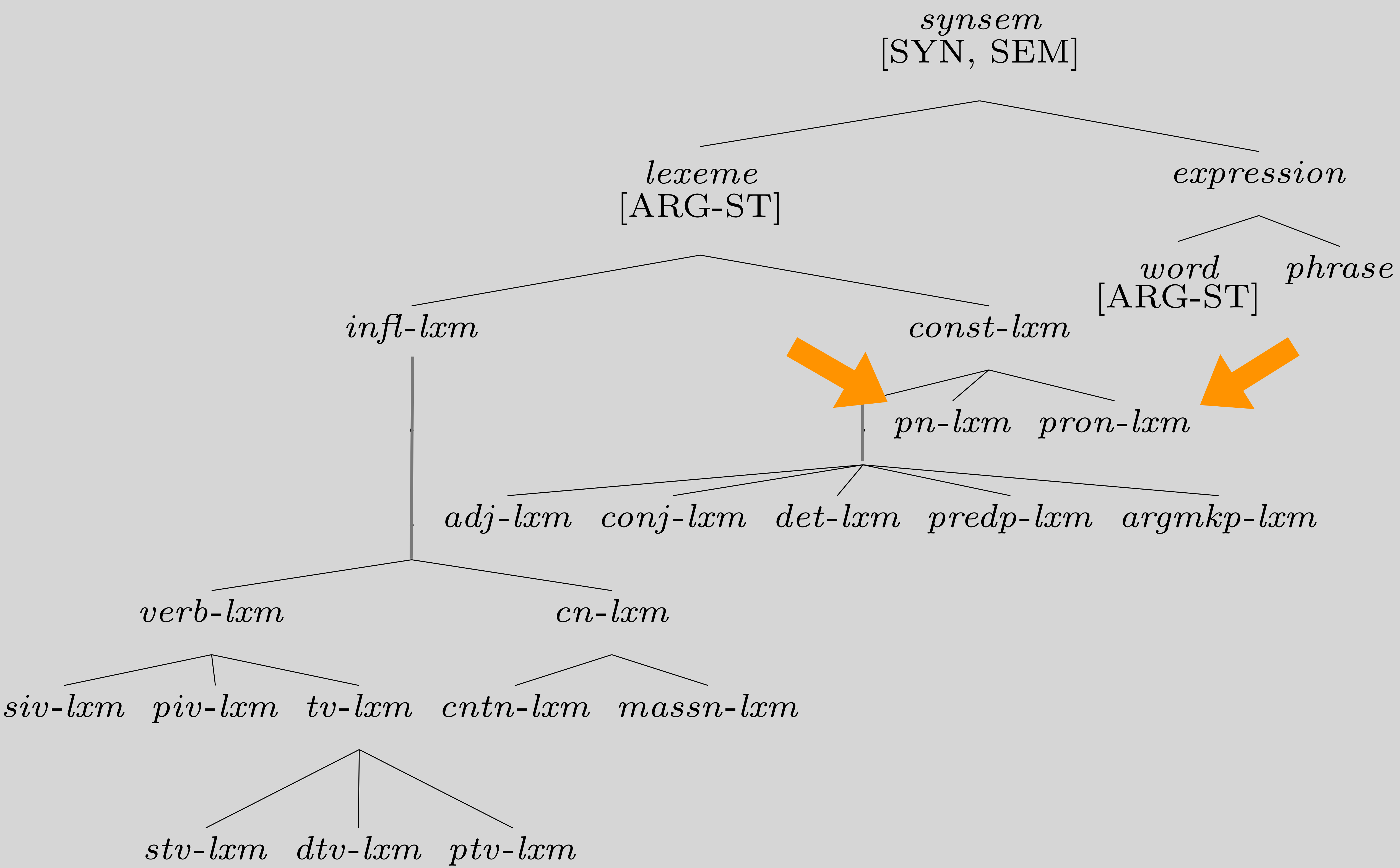
$$\textit{verb-lxm:} \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \textit{verb} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{MODE} & \text{prop} \end{bmatrix} \\ \text{ARG-ST} & / \left\langle \text{NP}, \ldots \right\rangle \end{bmatrix}$$

Subtypes of *verb-lxm*



- *verb-lxm*: [ARG-ST < NP, ... >]
  - *siv-lxm*: [ARG-ST < NP >]
  - *piv-lxm*: [ARG-ST < NP, PP >]
  - *tv-lxm*: [ARG-ST < NP, NP, ... >]
    - *stv-lxm*: [ARG-ST < NP, NP >]
    - *dtv-lxm*: [ARG-ST < NP, NP, NP >]
    - *ptv-lxm*: [ARG-ST < NP, NP, PP >]

Proper Nouns and Pronouns



Proper Nouns and Pronouns

$$\textit{pn-lxm}: \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{noun} & \\ \text{AGR} & \begin{bmatrix} \text{PER} & \text{3rd} \\ \text{NUM} & /\ \text{sg} \end{bmatrix} \end{bmatrix} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{MODE} & \text{ref} \end{bmatrix} \\ \text{ARG-ST} & /\ \langle\ \rangle \end{bmatrix}$$

$$\textit{pron-lxm}: \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \textit{noun} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{MODE} & /\ \text{ref} \end{bmatrix} \\ \text{ARG-ST} & \langle\ \rangle \end{bmatrix}$$

The Case Constraint

An outranked NP is [CASE acc].

• object of verb	✓
• second object of verb	✓
• object of argument-marking preposition	✓
• object of predicational preposition	(✓)

The Case Constraint, continued

An outranked NP is [CASE acc].

- Subjects of verbs
  - Should we add a clause to cover nominative subjects?
    - No.

    We expect them to leave. (Chapter 12)

    - Lexical rules for finite verbs will handle nominative subjects.
- Any other instances of case marking in English?
- Does it apply to case systems in other languages?

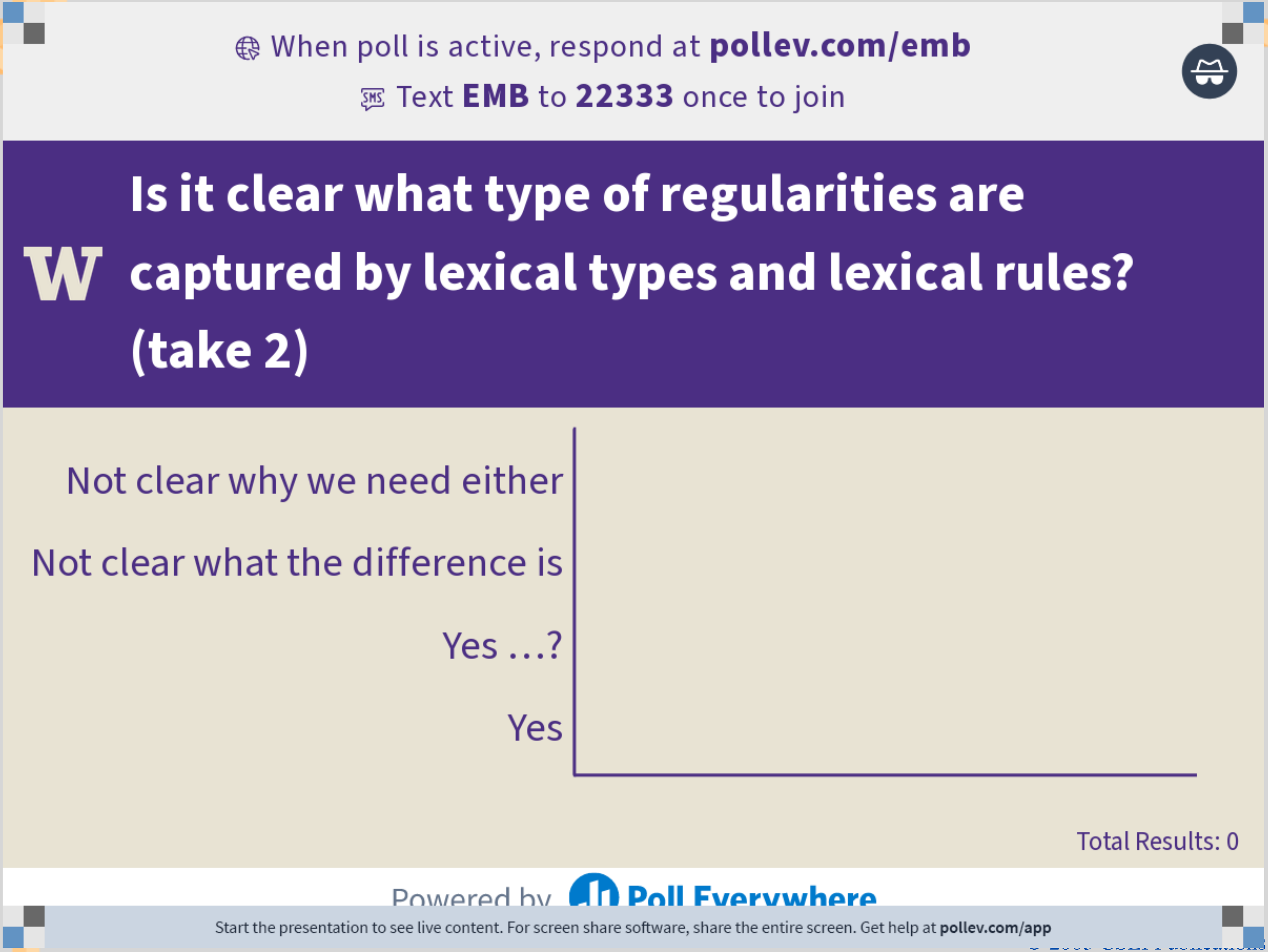
  No: The Case Constraint is an English-specific constraint.

Apparent redundancy

- Why do we need both the *pos* subhierarchy and lexeme types?
- *pos*:
    - Applies to words and phrases; models relationship between then
    - Constrains which features are appropriate (no AUX on *noun*)
- *lexeme*:
    - Generalizations about combinations of constraints

Lexical Types & Lexical Rules

- Lexemes capture the similarities among *run, runs, running,* and *run*.
- The lexical type hierarchy captures the similarities among *run, sleep,* and *laugh*, among those and other verbs like *devour* and *hand*, and among those and other words like *book*.
- Lexical rules capture the similarities among *runs*, *sleeps, devours, hands*,...



When poll is active, respond at **pollev.com/emb**

Text **EMB** to **22333** once to join

Is it clear what type of regularities are captured by lexical types and lexical rules? (take 2)

Not clear why we need either

Not clear what the difference is

Yes ...?

Yes

Total Results: 0

Powered by Poll Everywhere

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

Overview

- Motivation for lexical hierarchy
- Default inheritance
- Tour of the lexeme hierarchy
- The Case Constraint
- *pos* vs. *lexeme*
- Reading Questions

HW4 tips

- Ch 7 Problem 1:
  - Not grading you on the judgments, but on the sentences constructed and matching classification to the judgments
  - Be sure to keep the same verb + preposition pair
- Ch 8 grammar summary is in Ch 9

RQs: Defeasible constraints

- Now that feature values can have "default values" with the / notation, this means that a missing feature in a matrix could mean any of the following:
  - It's underspecified;
  - It's omitted for brevity, or
  - It's falling back to the default value.
- How do we tell which is which?

RQs: Defeasible constraints

- It seems that we only mark whether a constraint is defeasible or not using "/". Perhaps this will be mentioned in 8.6-8.8, but I was wondering if this rule is ever extended to specify in specifically what cases a constraint can be overridden?

- I'm curious about if having default constraints for a lexical type means that we don't need to specify them in lexical entries that are of that type. Take the default constraints on type lexeme MOD /< >: does this mean that any lexical entry does not have to include MOD <> to be considered fully specified?

- In grammar design, how do we decide when to write defeasible constraints?

RQs: lex entries/lex sequence

- Can you explain the difference between lexical entries and lexical sequences more?
- What is the difference between a lexical entry and a lexical sequence? Does a family of lexical sequences describe the different forms of the same lexeme?

RQs: phrase, word, lexeme

- As of this chapter, are we officially eliminating 'phrase' and 'word' from our trees and lexical entries and replacing them with pn-lxm, dtv-lxm, etc? As a result, does this mean that we do not need to rewrite the information of a given constraint if it has not been overruled? For example, the constraint for pn-lxm states it is MODE ref so we can omit MODE ref.

RQs: phrase, word, lexeme

- Why is lexeme not of the type expression? It feels like word should be a subtype of lexeme but it is not organized this way.
- Would it be possible to build a "tree" for a sentence pattern rather than a fully specified sentence, using lexemes as the leaves?

RQs: SPR on modifiers

- The way that predp-lxm and adj-lxm specify both the MOD and SPR values imply that it takes both the head-specifier rule and the head-modifier rule to attach a modifier to a word. Is it actually possible to somehow apply both rules together?

RQs: X, Y, Z

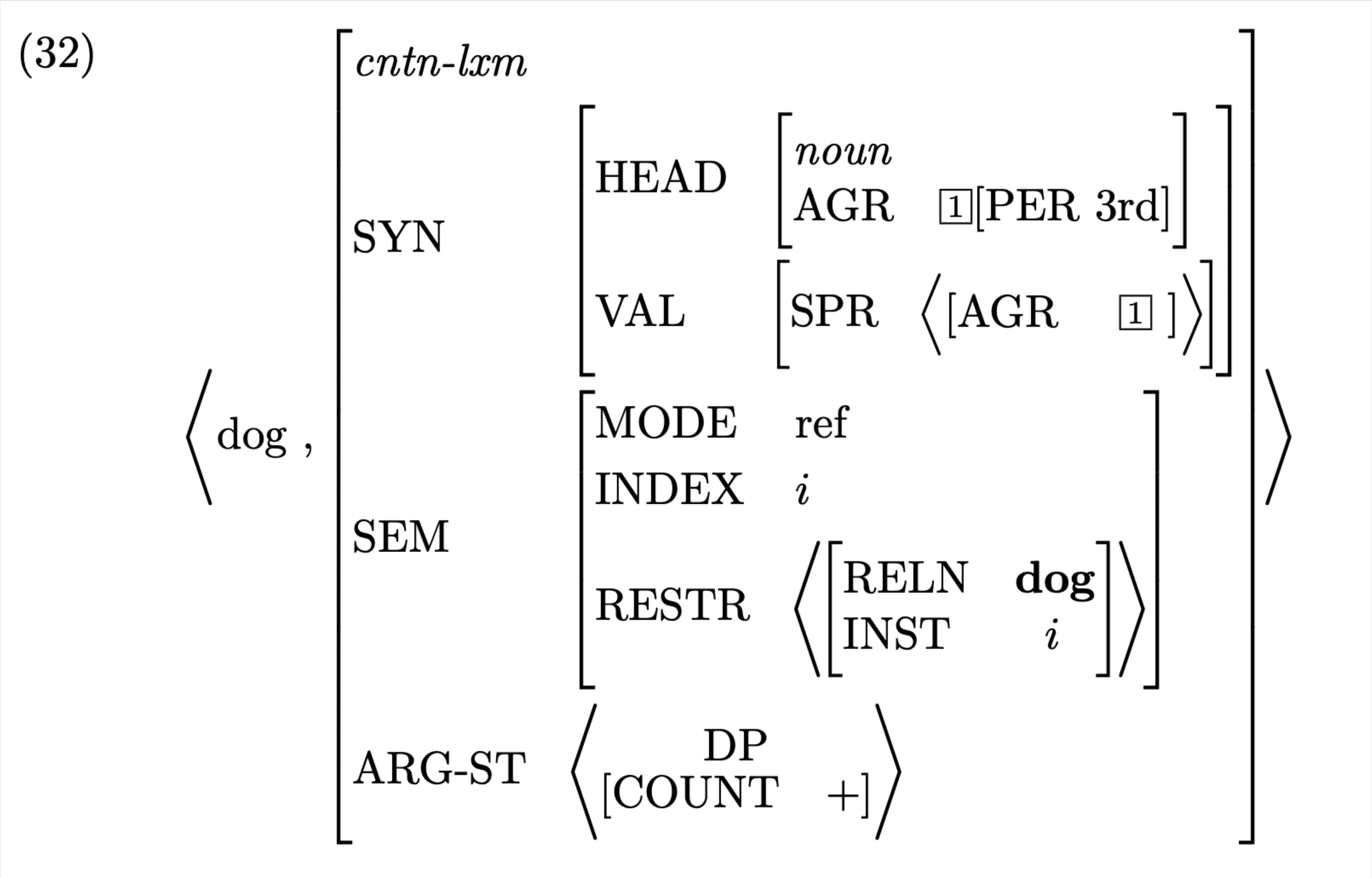
- In the following tree (and in several ARG-ST lists), why are some elements of ARG-ST shown as X, Y, etc? Why are we not using NP, PP, etc. directly?

RQs: lexical ambiguity

- Instead of having around live a double life as a *predp-lxm* and *argmkp-lxm*, couldn't we create a supertype of these two types for all prepositions and just underspecify around as belonging to neither of these two?

RQs: ARP

- According to the Argument Realization Principle, AGR-ST is the sum of the SPR value and the COMPS value. So why is the SPR value different from the first element in AGR-ST in (32)?



(32)

$$\left\langle \text{dog}\ ,\ \left[\begin{array}{ll}
\textit{cntn-lxm} & \\
\text{SYN} & \left[\begin{array}{ll}
\text{HEAD} & \left[\begin{array}{ll} \textit{noun} & \\ \text{AGR} & \boxed{1}[\text{PER 3rd}] \end{array}\right] \\
\text{VAL} & \left[\text{SPR}\ \left\langle [\text{AGR}\ \ \boxed{1}\ ] \right\rangle\right]
\end{array}\right] \\
\text{SEM} & \left[\begin{array}{ll}
\text{MODE} & \text{ref} \\
\text{INDEX} & i \\
\text{RESTR} & \left\langle \left[\begin{array}{ll} \text{RELN} & \textbf{dog} \\ \text{INST} & i \end{array}\right] \right\rangle
\end{array}\right] \\
\text{ARG-ST} & \left\langle \begin{array}{c} \text{DP} \\ [\text{COUNT}\ \ +] \end{array} \right\rangle
\end{array}\right] \right\rangle$$

RQs: CASE

- As a speaker of a language with a fully developed all-encompassing case system, I find our grammar's insistance on case being a feature of all nouns to be at the very least strange.

RQs: Implementation

- I can imagine a grammar with an untenable amount of word classes. How many word classes are there in a good grammar. I was surprised to read about a class for sports teams and a class for mountain ranges.
- What kind of variation do we see in the number of word classes across languages?
- How are lexical entries used in practice in a computational setting? Are lexical entries formed ahead of time or are they usually built in context?