Ling 566
Oct 28, 2021

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

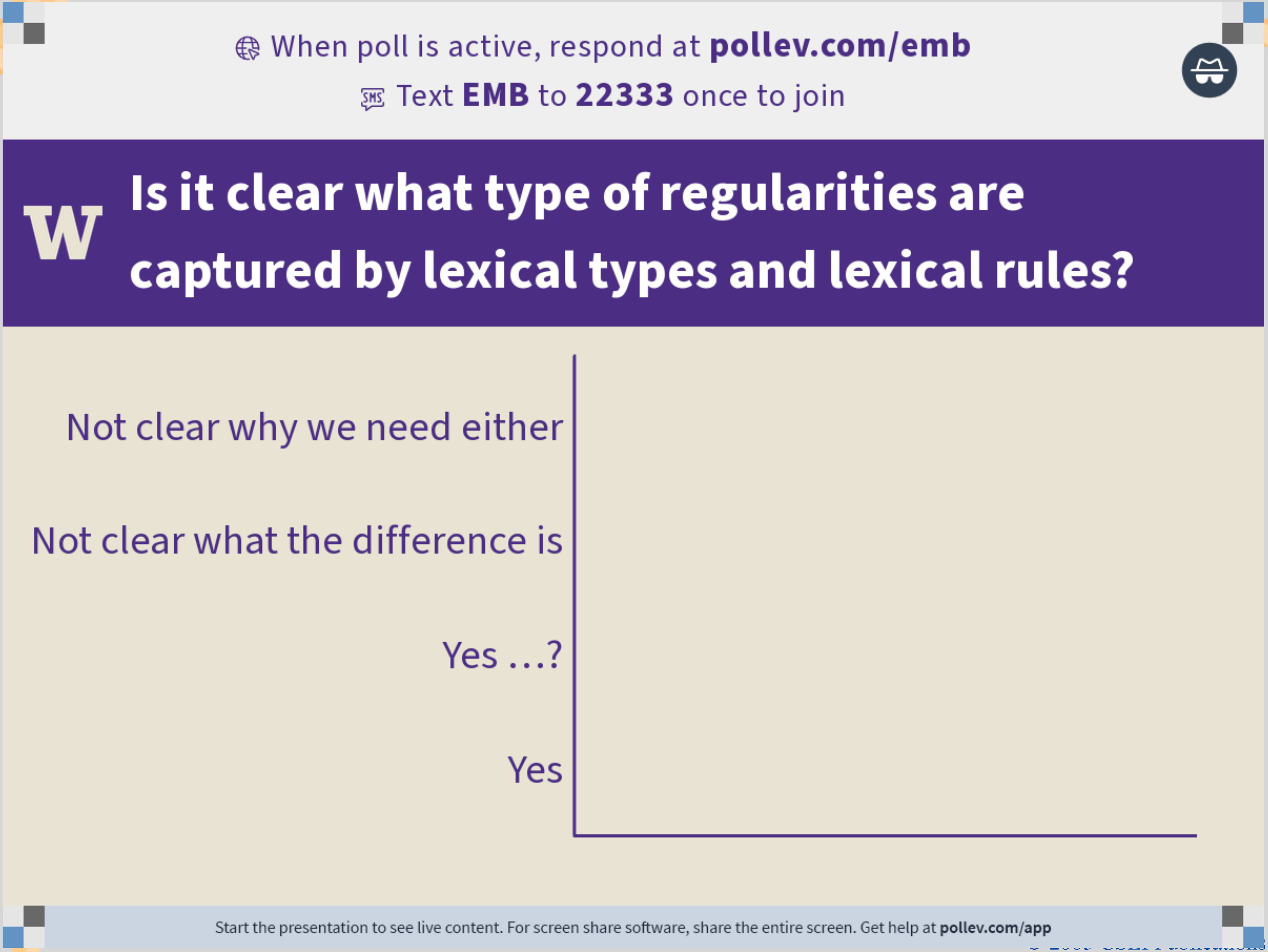
- Lexemes capture the similarities among *run, runs, running,* and *run*.

- 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

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’

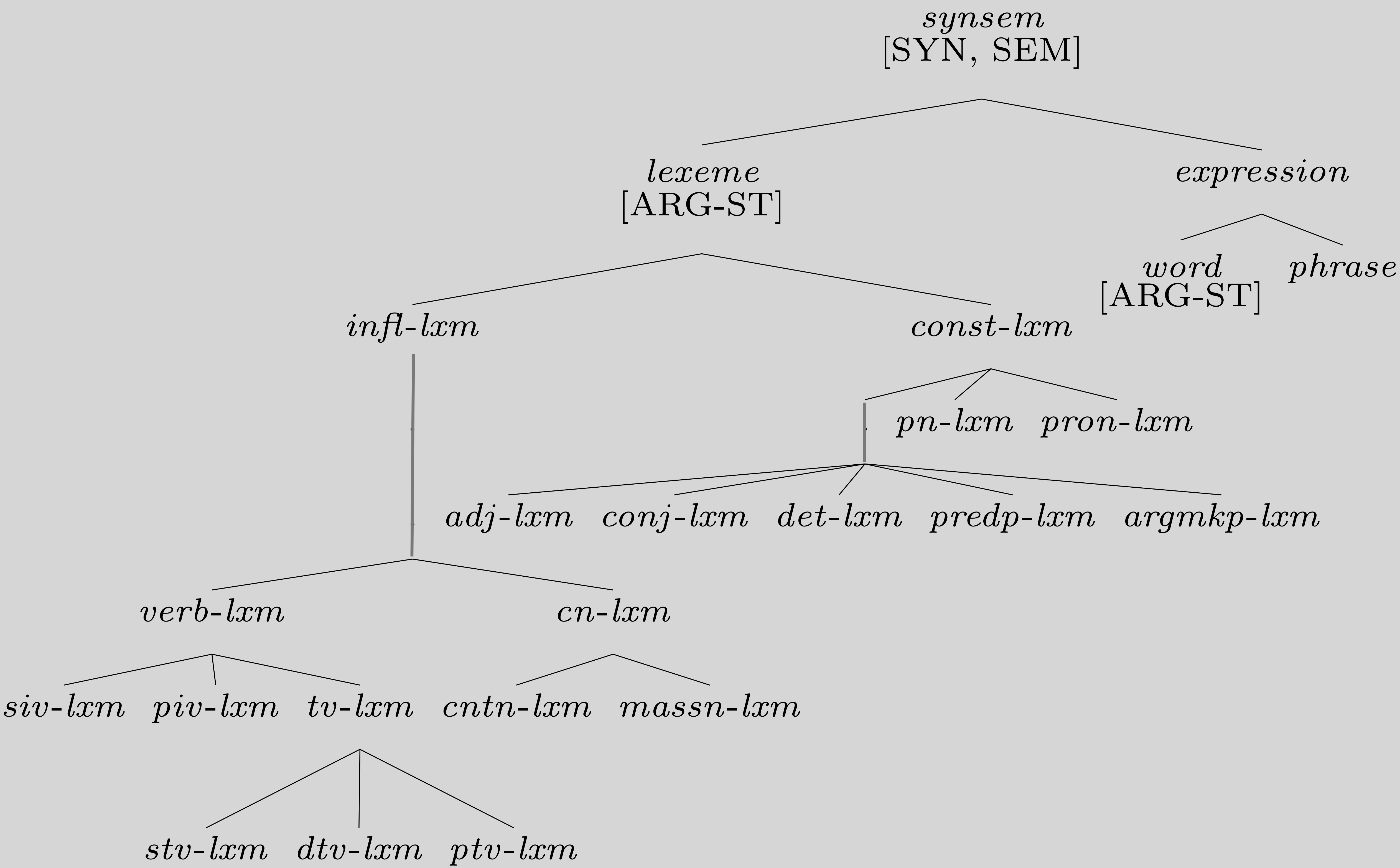
Question on Default Inheritance

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*

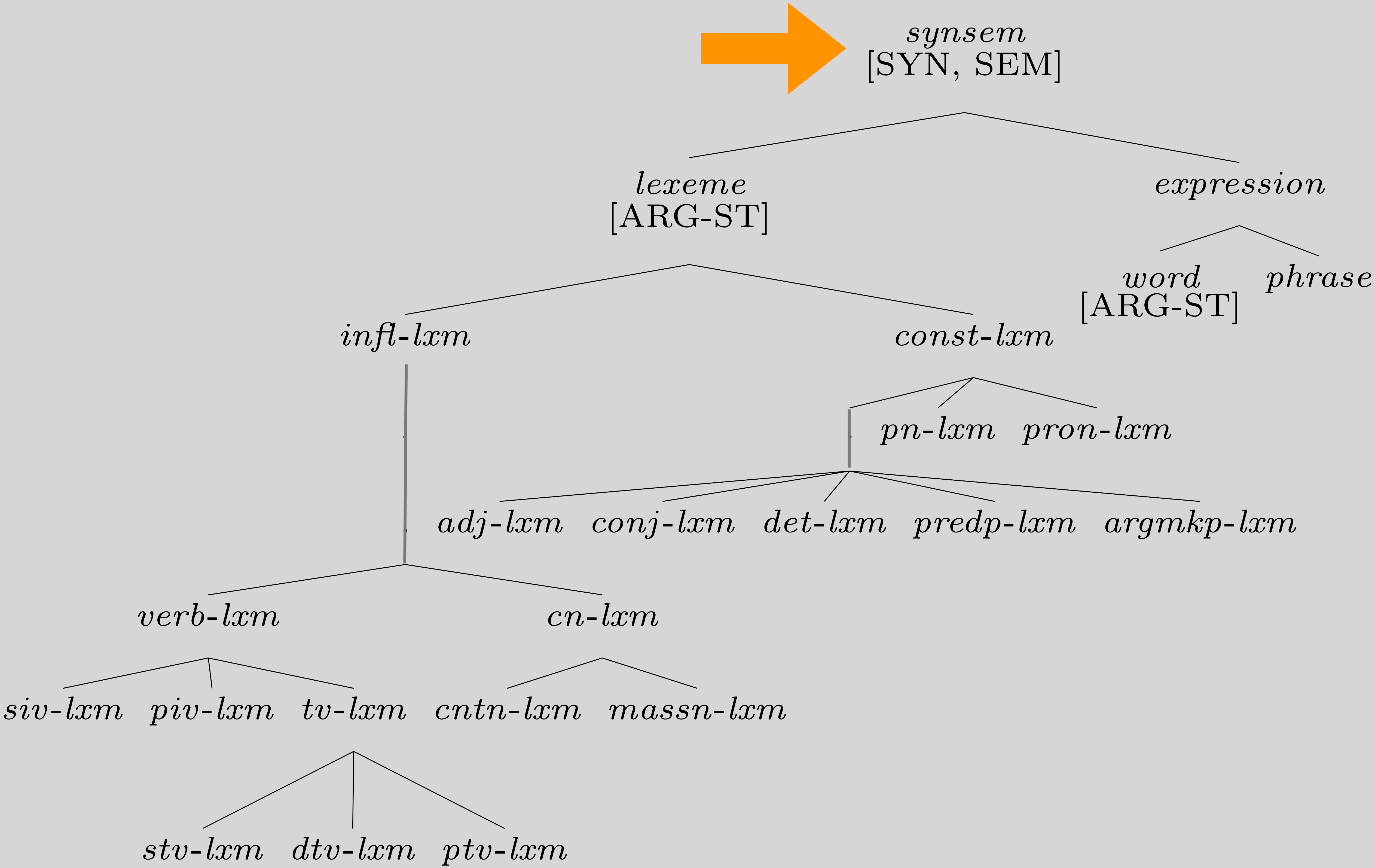
siv-lxm *piv-lxm* *tv-lxm* *cntn-lxm* *massn-lxm*

stv-lxm *dtv-lxm* *ptv-lxm*

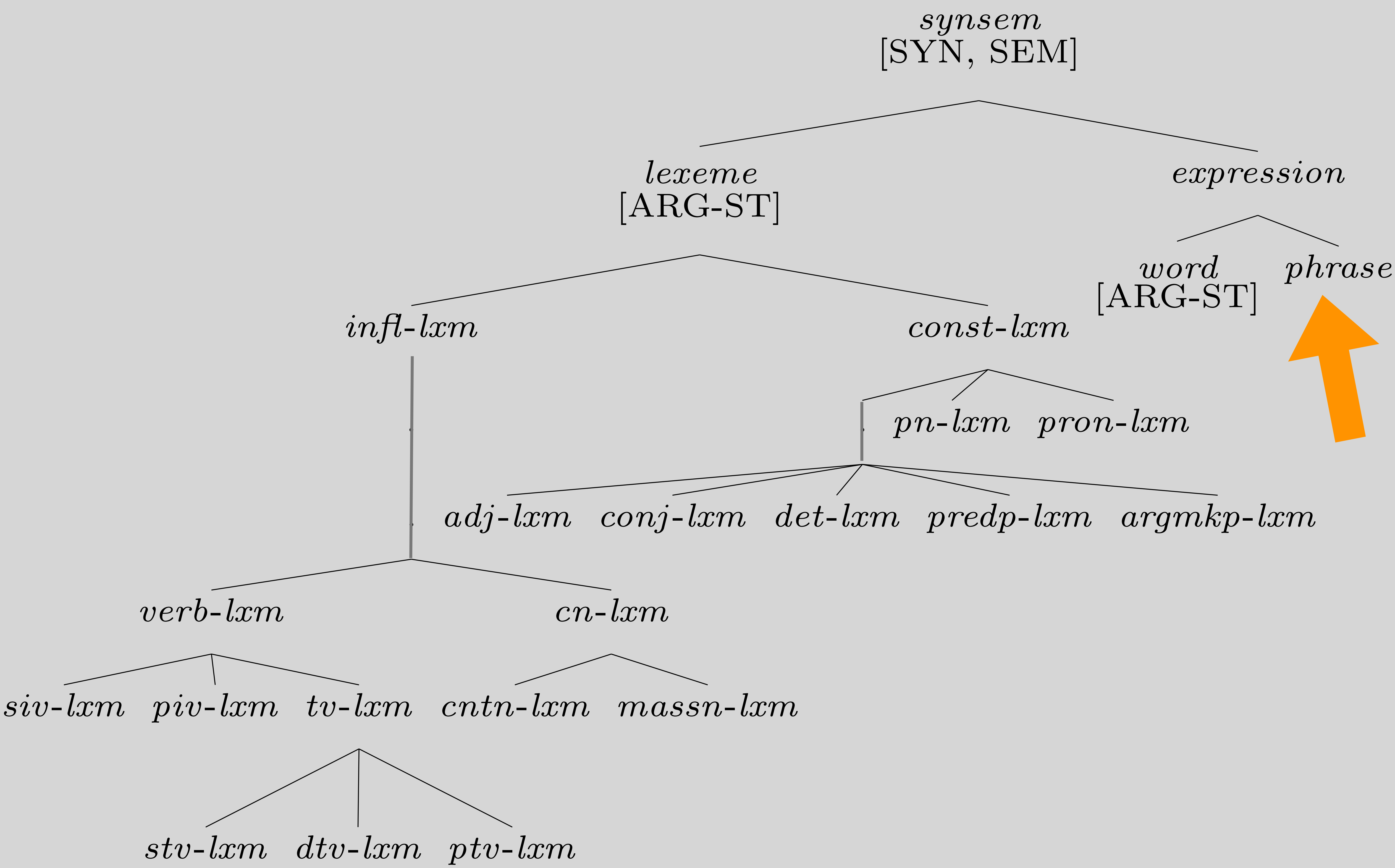
Functions of Types

- 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*



synsem
[SYN, SEM]

lexeme
[ARG-ST]

expression

word
[ARG-ST]

phrase

infl-lxm

const-lxm

pn-lxm *pron-lxm*

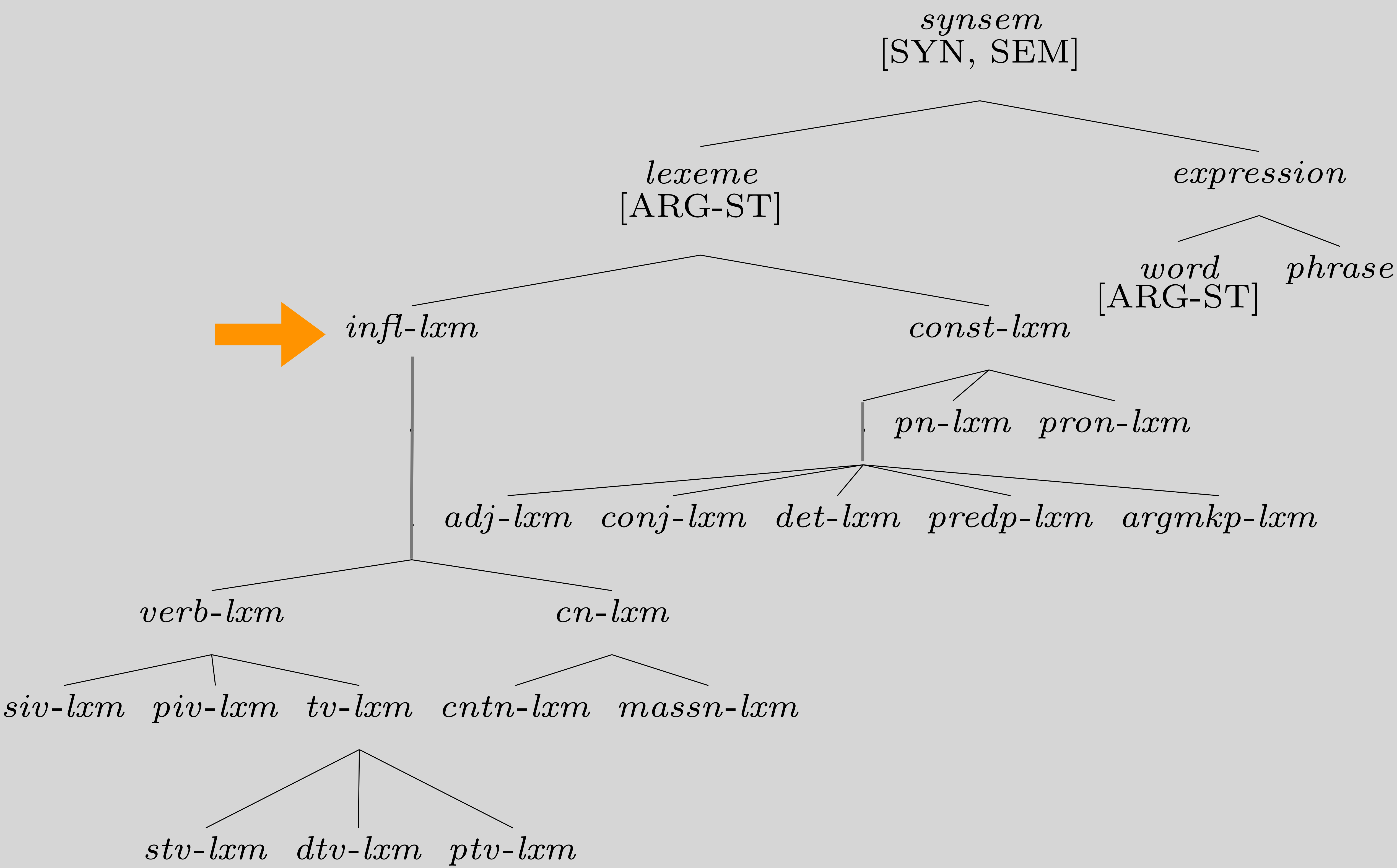
adj-lxm *conj-lxm* *det-lxm* *predp-lxm* *argmkp-lxm*

verb-lxm *cn-lxm*

siv-lxm *piv-lxm* *tv-lxm* *cntn-lxm* *massn-lxm*

stv-lxm *dtv-lxm* *ptv-lxm*

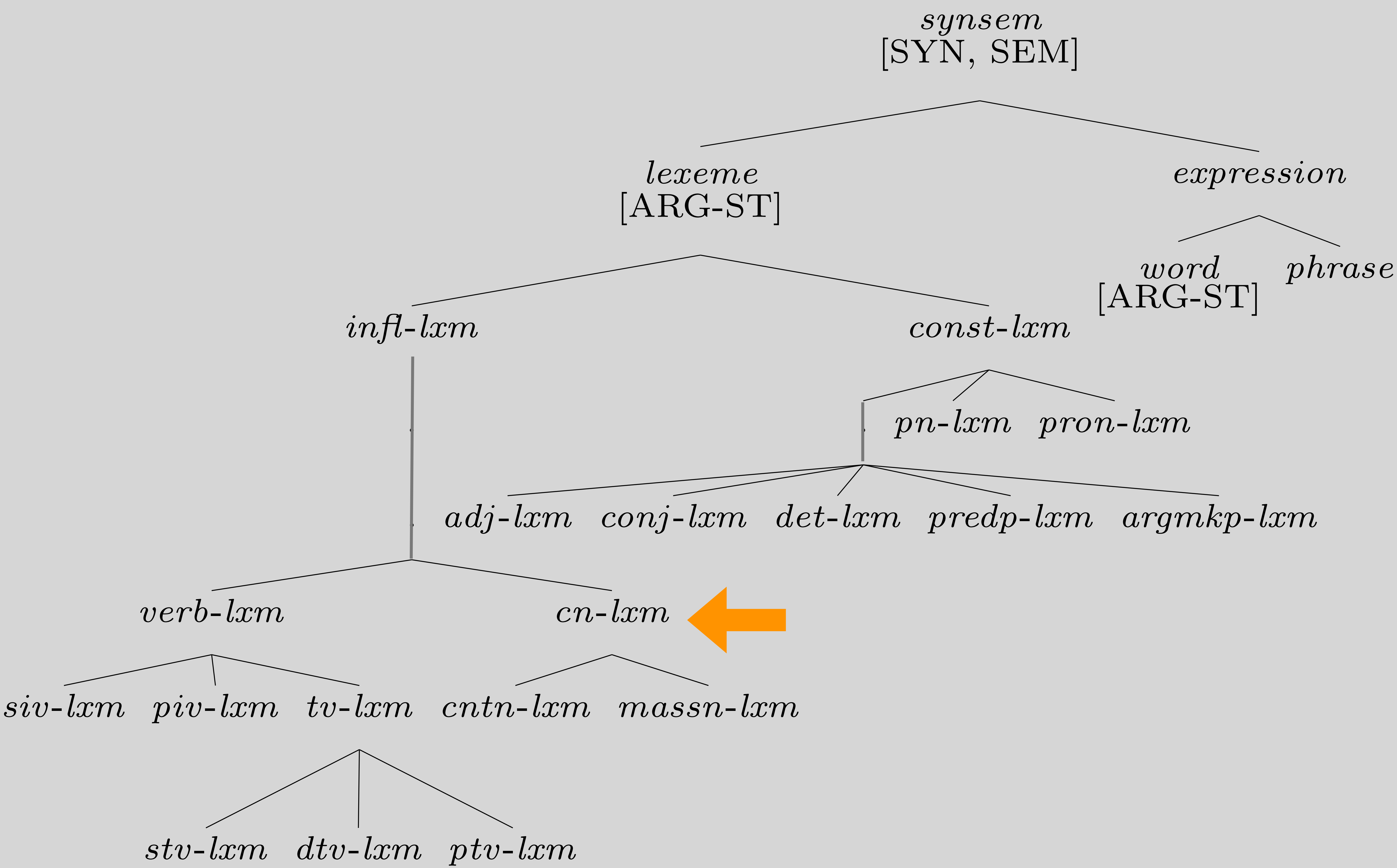
A Constraint on *infl-lxm*: the SHAC



A Constraint on *infl-lxm*: the SHAC

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

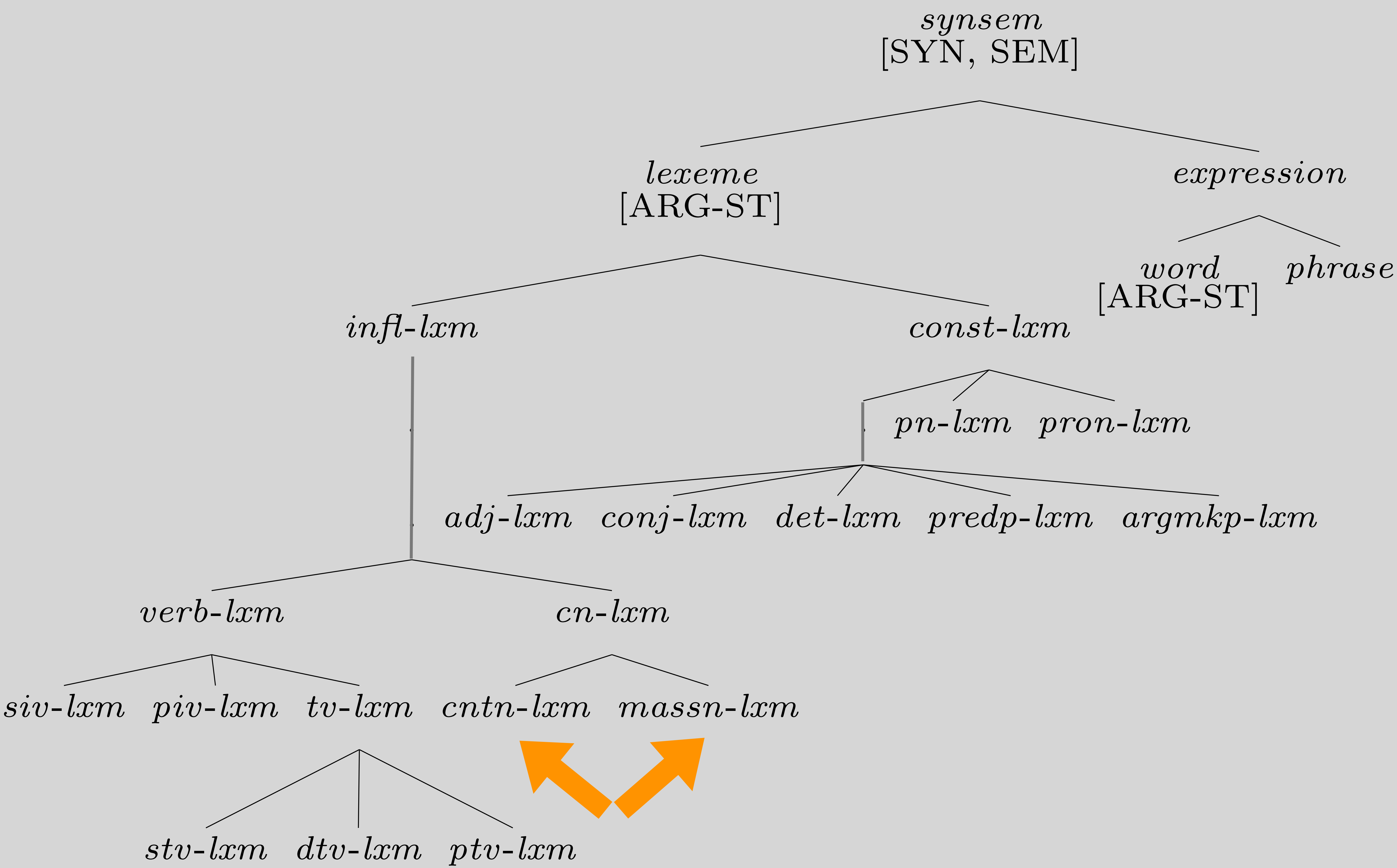
Constraints on *cn-lxm*



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

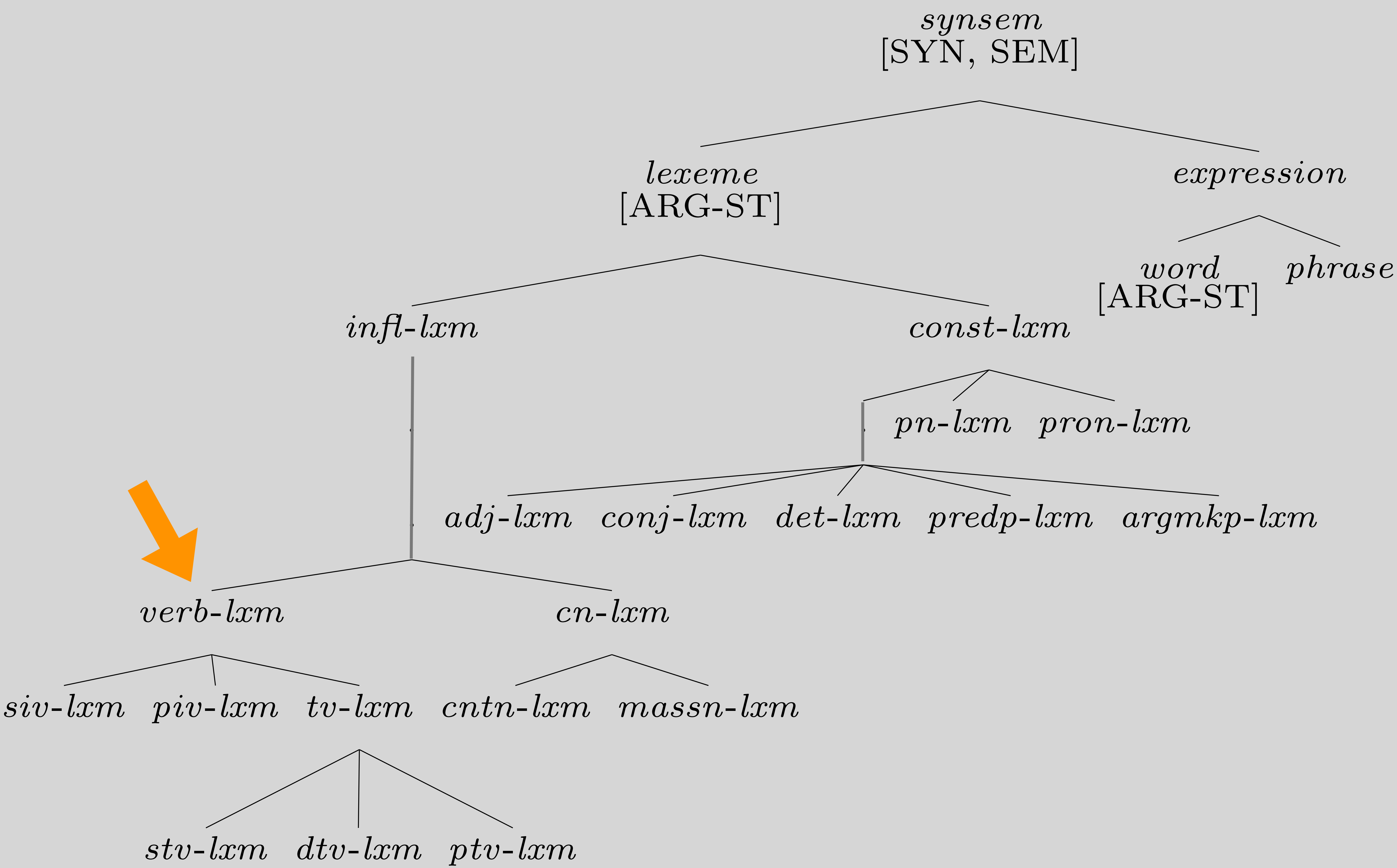


Formally Distinguishing Count vs. Mass Nouns

$$\textit{cntn-lxm} : \left[\text{SYN} \quad \left[\text{VAL} \quad \left[\text{SPR} \quad \langle\ [\text{COUNT } +]\ \rangle\right]\right]\right]$$

$$\textit{massn-lxm} : \left[\text{SYN} \quad \left[\text{VAL} \quad \left[\text{SPR} \quad \langle\ [\text{COUNT } -]\ \rangle\right]\right]\right]$$

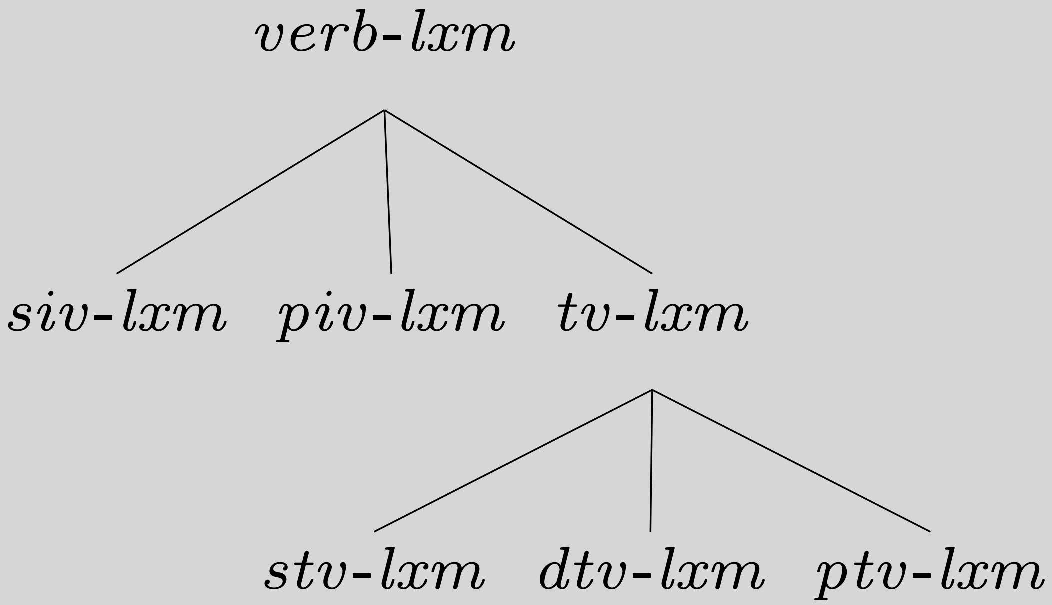
Constraints on *verb-lxm*



Constraints on *verb-lxm*

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

Subtypes of *verb-lxm*



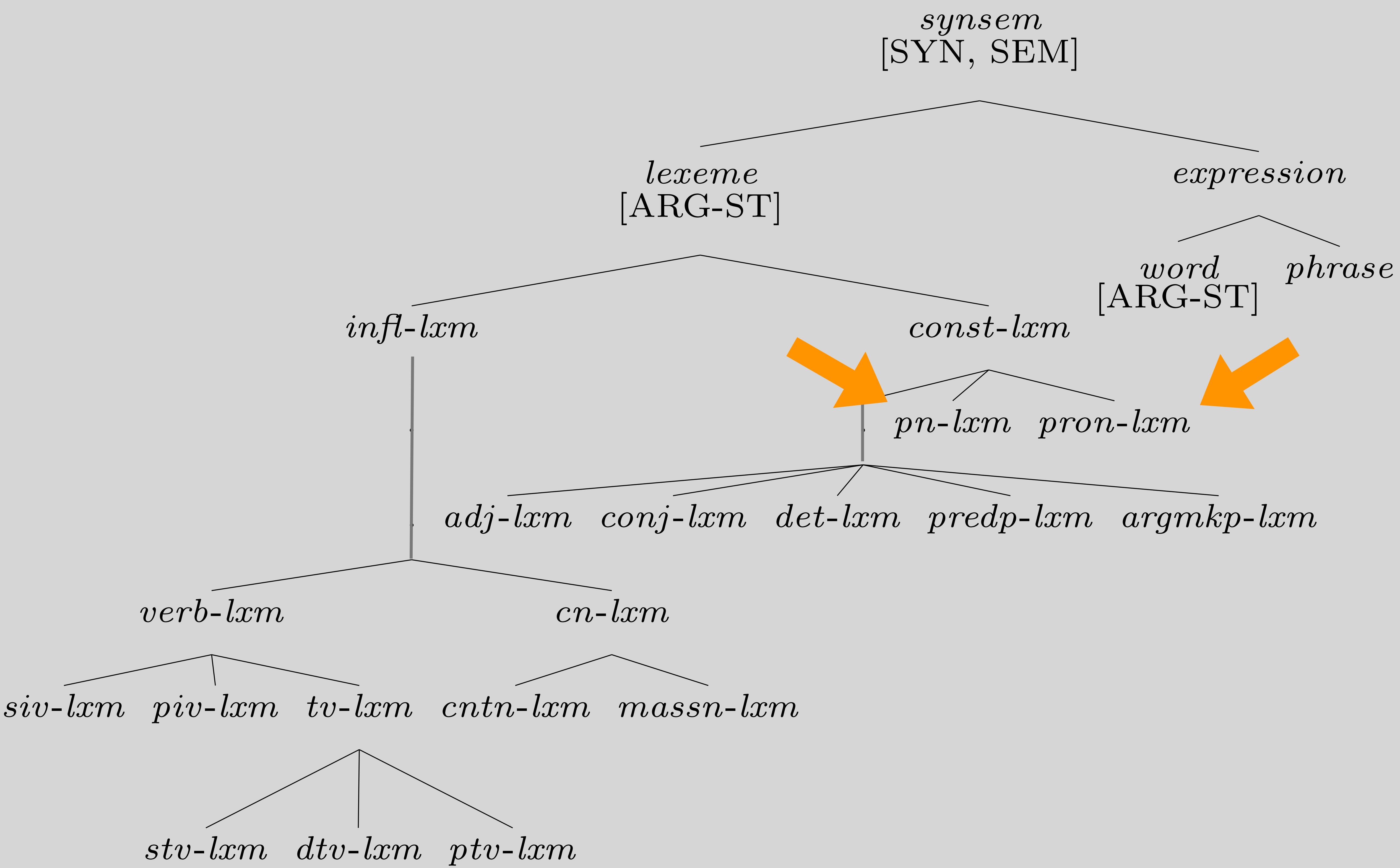
verb-lxm

siv-lxm *piv-lxm* *tv-lxm*

stv-lxm *dtv-lxm* *ptv-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

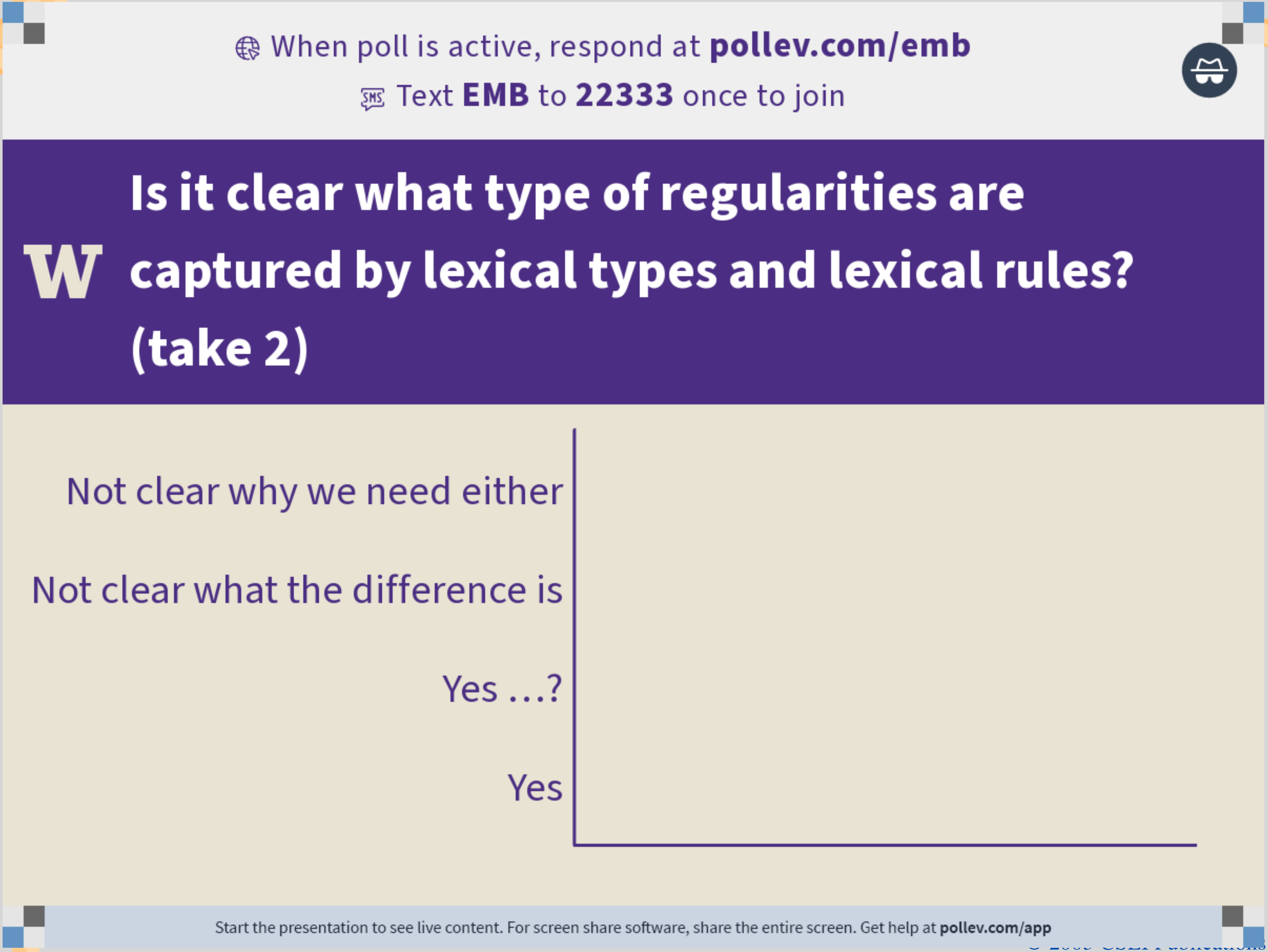
- Why do we need both the *pos* subhierarchy and lexeme types?
- *pos*:
  - Applies to words and phrases; models relationship between then
  - Constrains which features are appropriate (no AUX on *noun*)
- *lexeme*:
  - Generalizations about combinations of constraints

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

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

Reading Questions

- I'd like a bit more clarification between a lexeme, a lexical entry, and a word. Are lexical entries all lexemes, and then anything on a tree is a word?
- Can you compare the lexical sequences, lexical families (family of lexical sequences), lexical entry and lexeme against (next to) each other?
- I understood that the lexical entries license infinitely many resolved feature structures. But what does it mean that infinitely many lexical sequences satisfy a lexical entry? I am unclear on what "sequences" means in this context.
- What is the difference between "lexeme" and "lexical item"?

Reading Questions

- Does the *lexeme* hierarchy fit into the *feat-struc* hierarchy?
- Would a word ever fall into more than one subtype such that it is in more than one branch of the lexeme tree? The structure is very elegant, but would we have a humongous lexeme tree for languages with many exceptions?

Reading Questions

- In (45a), *adj-lxm* has its SPR as <X>, and ARG-ST as <NP,...>. Does that imply its SPR is <NP>? Meanwhile, its MOD is also <[HEAD *noun*]>. Does that mean an adj modifies an NP and specifies another NP? Could you elaborate that with an example?

- In the constraints for *cn-lxm*, *cntn-lxm* and *massn-lxm*, why was ARG-ST chosen as the best location for the constraints related to specifiers rather than SPR?

Reading Questions

- Are the letters X, Y, and Z throughout this section just placeholders for elements that may appear in the lists for those features within a potential tree?

Reading Questions

- What makes a constraint inviolable? How do we distinguish them from defeasible constraints (other than looking for the "/" symbol)?
- How are we determine whether a constraint is defeasible or not? Are there any systematic ways to tell whether a constraint is defeasible or not? The defeasible specifications / conflicting specifications / inviolable specifications thing seem a bit not "general" enough to me...

Reading Questions

- On p. 234, it is stated that certain specifications are defeasible and they can be overridden by conflicting specification. Does it mean that the specifications in the lower position of the type hierarchy have greater influence on the leaves of the type hierarchy?

Reading Questions

- "As desired, the SHAC applies only to verbs and to common nouns. Notice that the SHAC is not a defeasible constraint." - Why is it not a defeasible constraint? Because it applies to the "supertype" *infl-lxm*?

- I'm also not intuiting why SHAC is a constraint on *infl-lxm*. Is it because inflection on the verb carries the relevant pieces of morphosyntactic information for agreement? How would this work for languages that have little, if any, inflectional morphology (e.g. Yoruba)?

Reading Questions

- The SHAC is now a constraint on *infl-lxm*, but *const-lxm* doesn't seem to have a similar generalized constraint that applies to it, instead several specific constraints based on what it is. Why is this? Is it because *const-lxms* are all too different?

Reading Questions

- Why didn't we structure the lexicon like this earlier? We know we have to account for these different types of lexemes at some point, but doesn't it make more sense to start with a robust lexicon and build rules and principles from there? Is the motivation for including this change in this section of the book pedagogical or is there a logical consistency with the construction of the grammar?