Ling 566
Oct 29, 2020

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*,...

Poll!

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?

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

- That is, default constraints are 'pushed down'

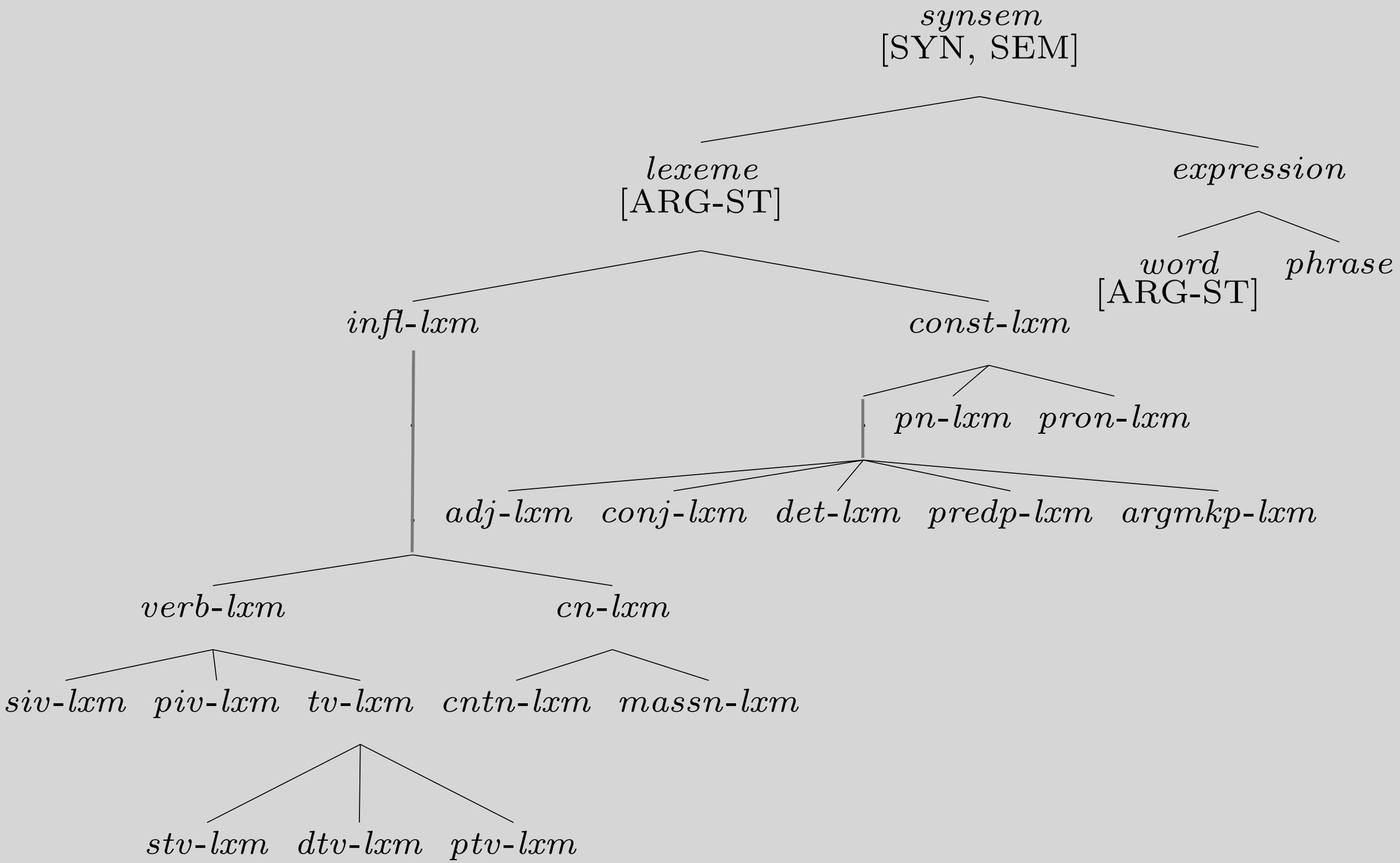
Question on Default Inheritance

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.

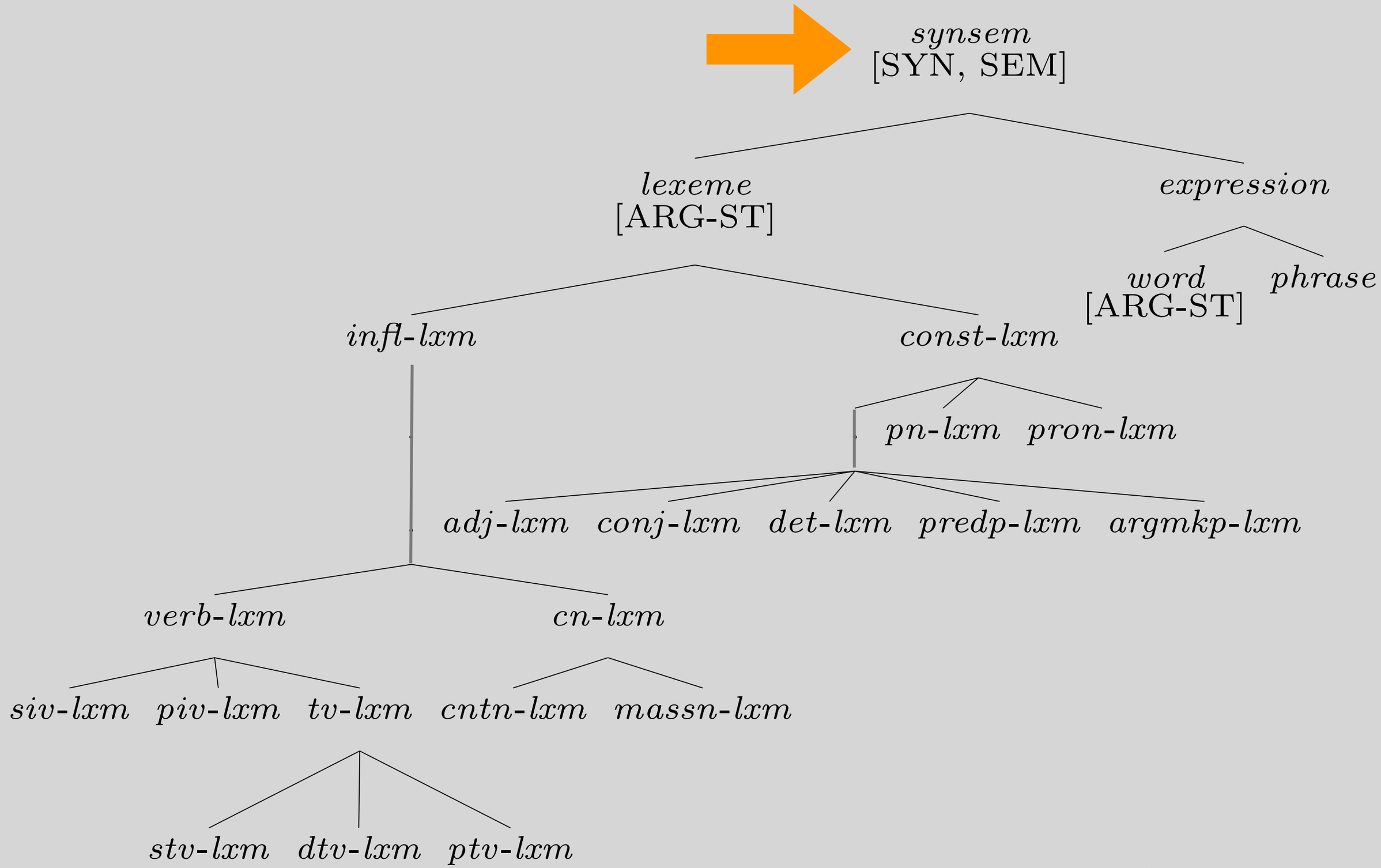
Our Lexeme Hierarchy



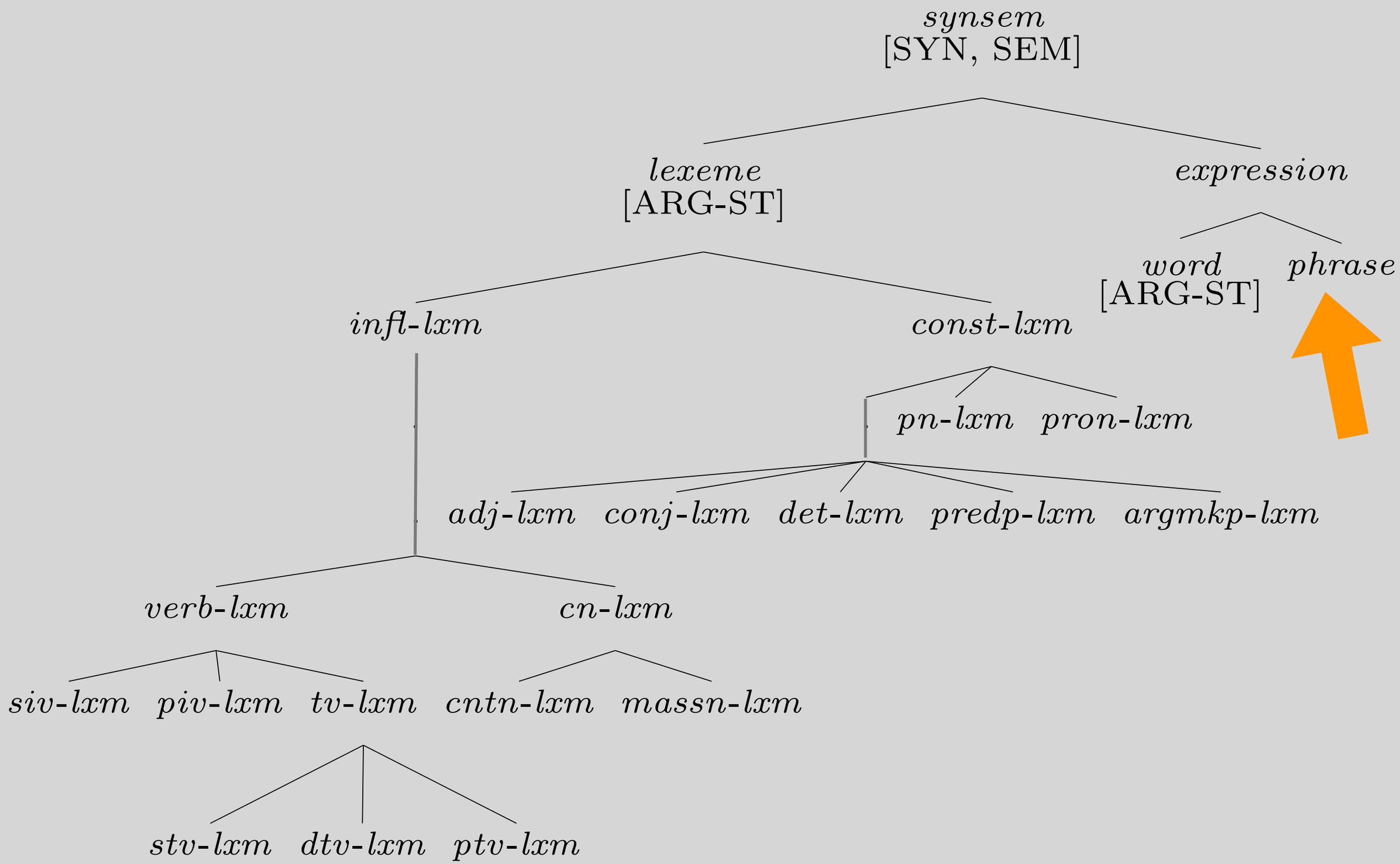
Functions of Types

- 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

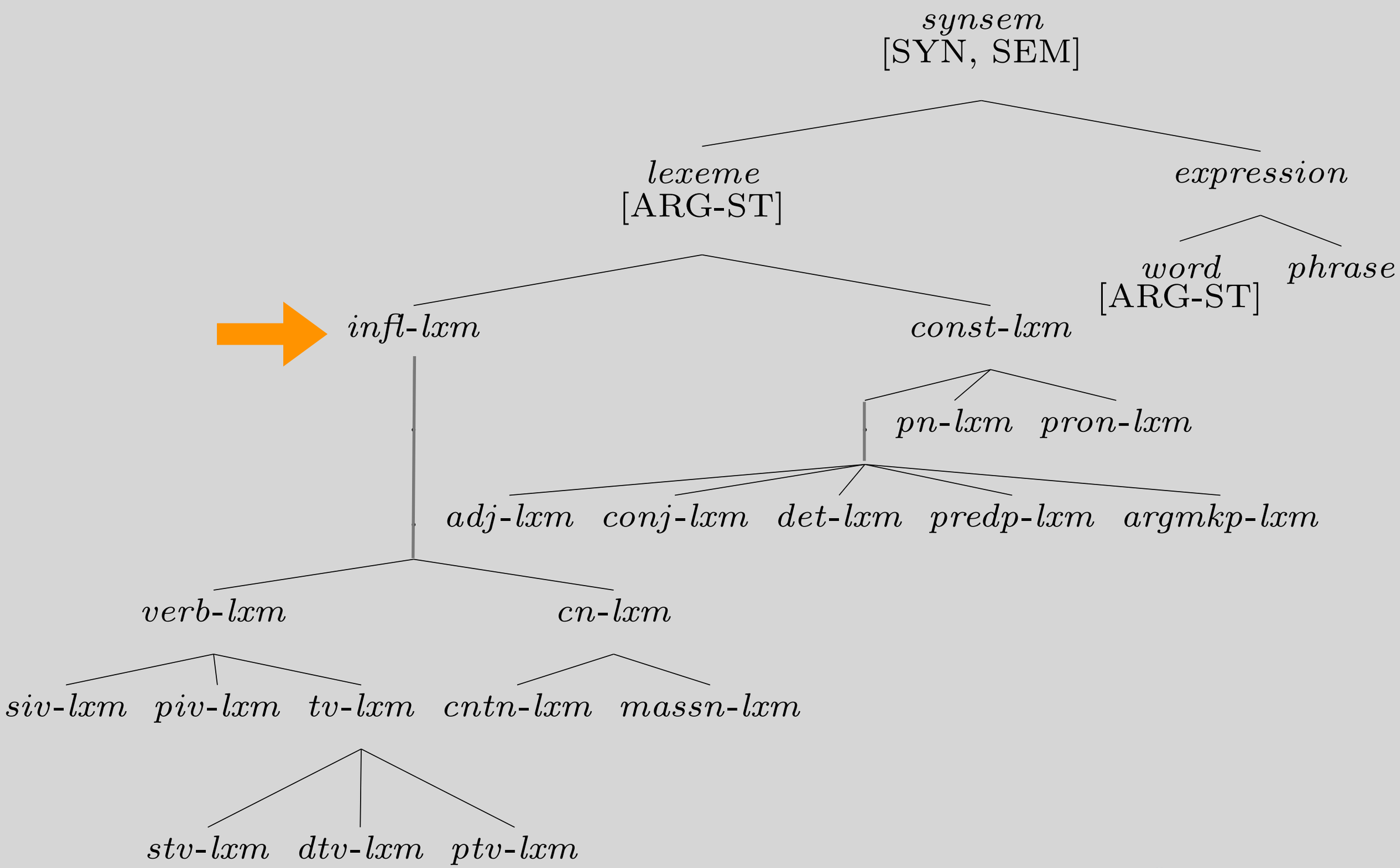
Every *synsem* has the features SYN and SEM



No ARG-ST on *phrase*



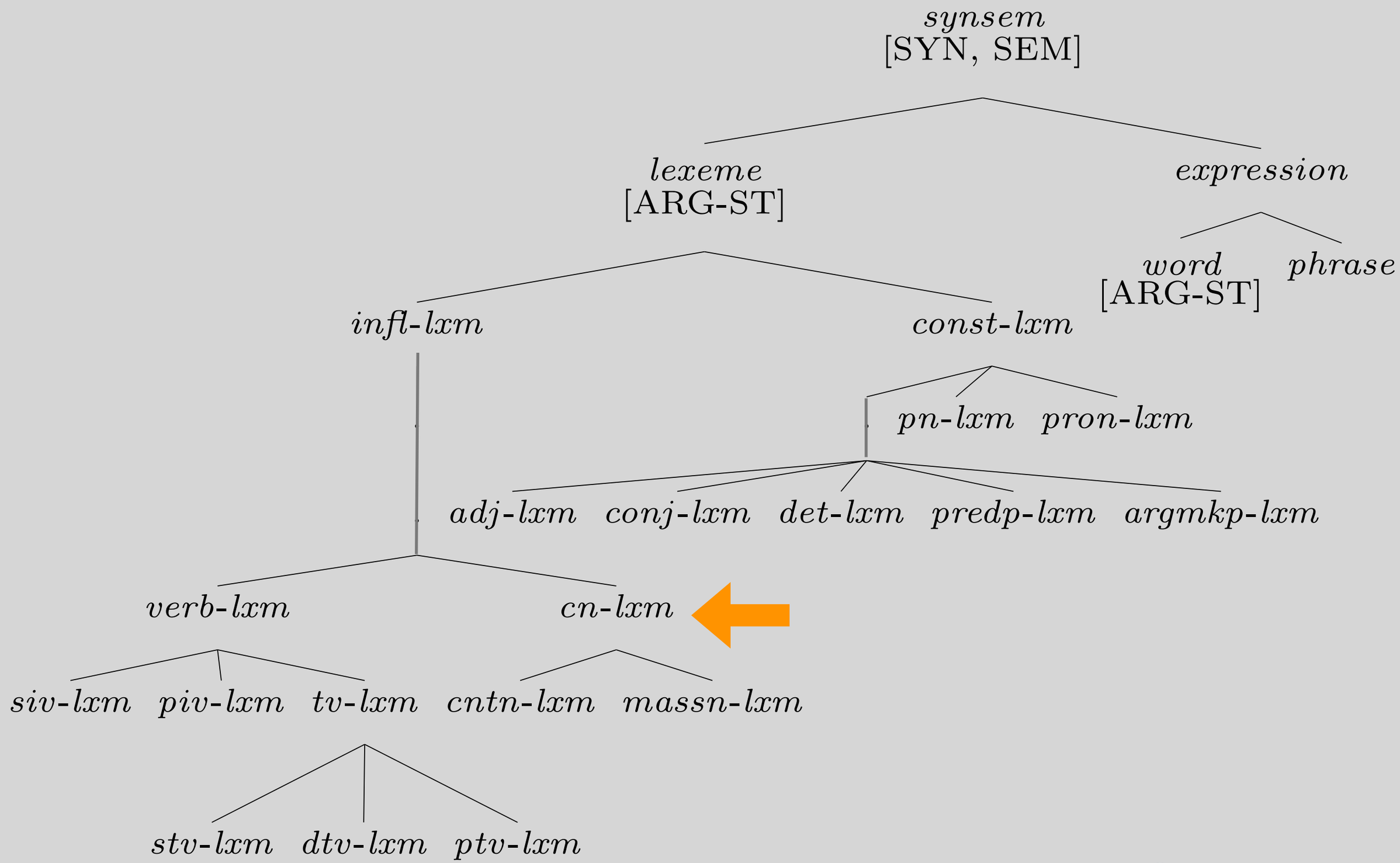
A Constraint on *infl-lxm*: the SHAC



A Constraint on *infl-lxm*: the SHAC

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

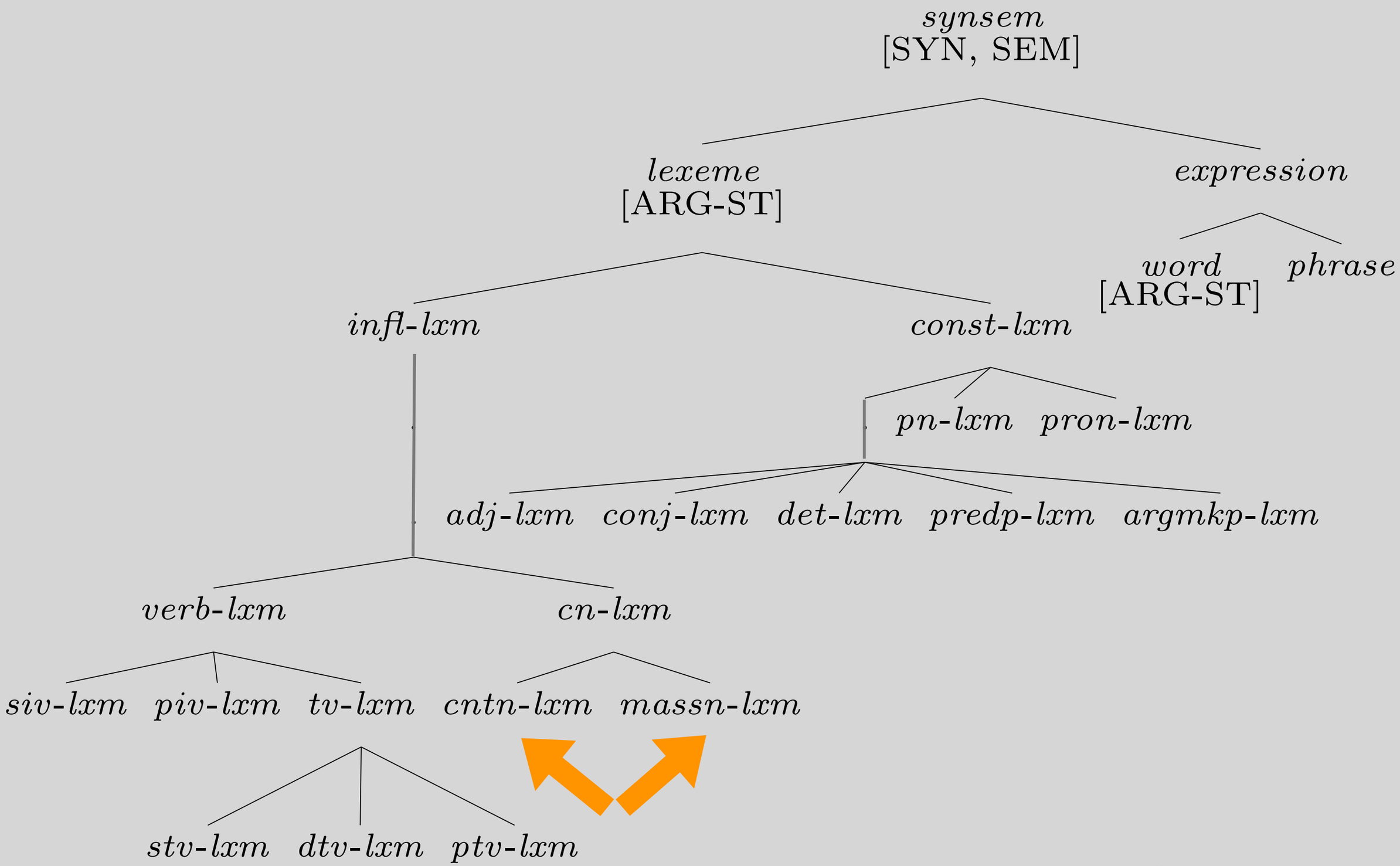
Constraints on *cn-lxm*



Constraints on *cn-lxm*

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

Formally Distinguishing Count vs. Mass Nouns

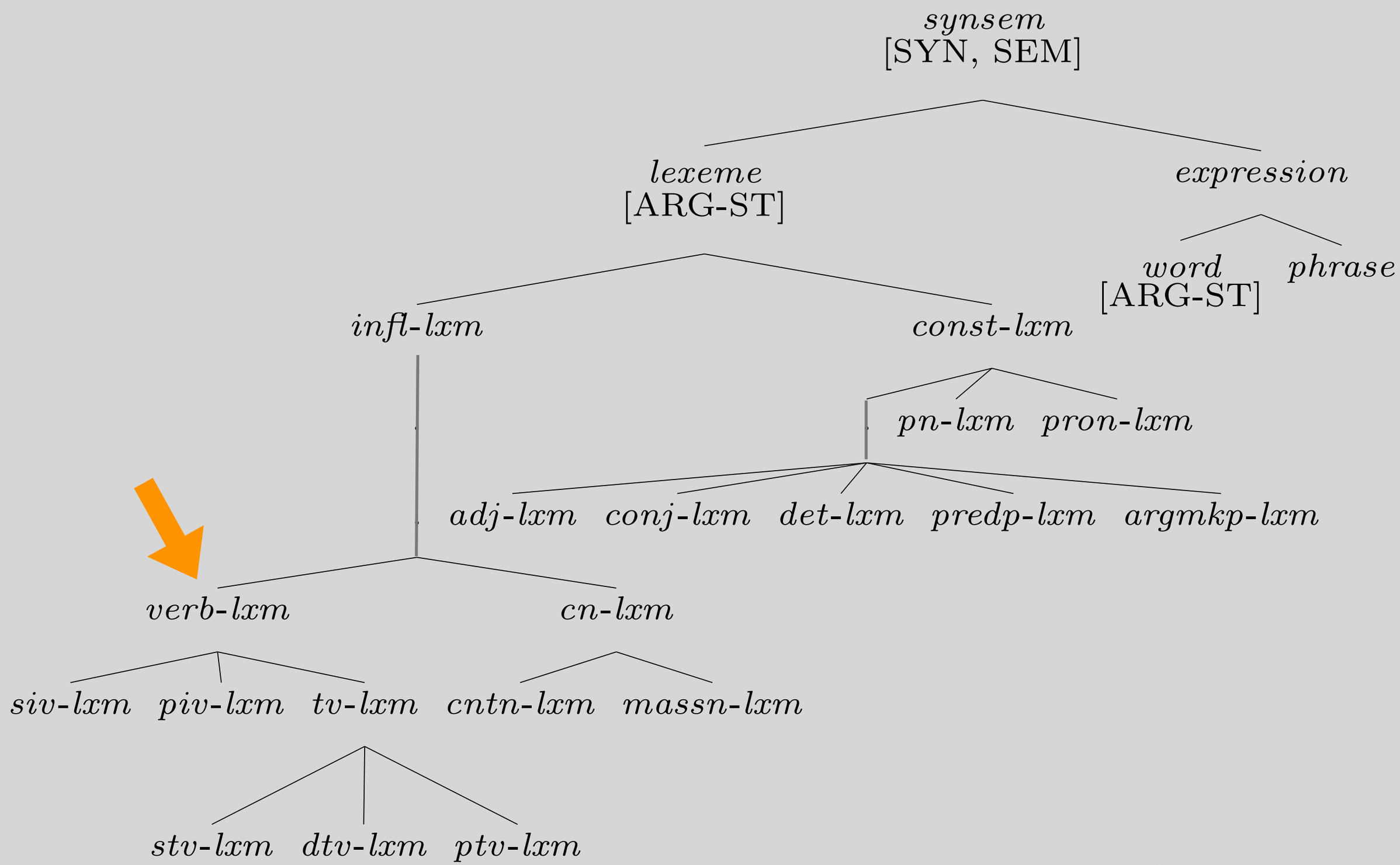


Formally Distinguishing Count vs. Mass Nouns

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

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

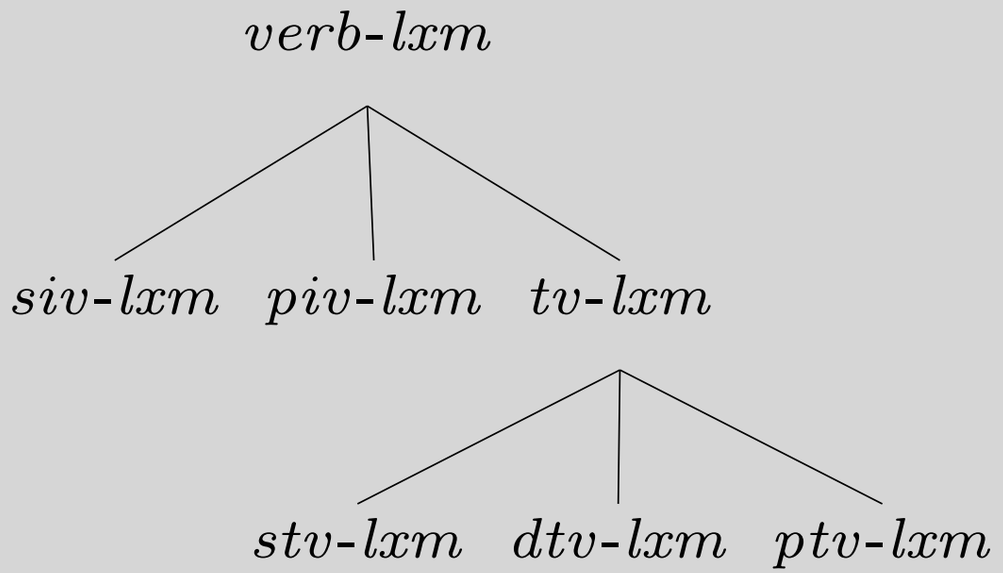
Constraints on *verb-lxm*



Constraints on *verb-lxm*

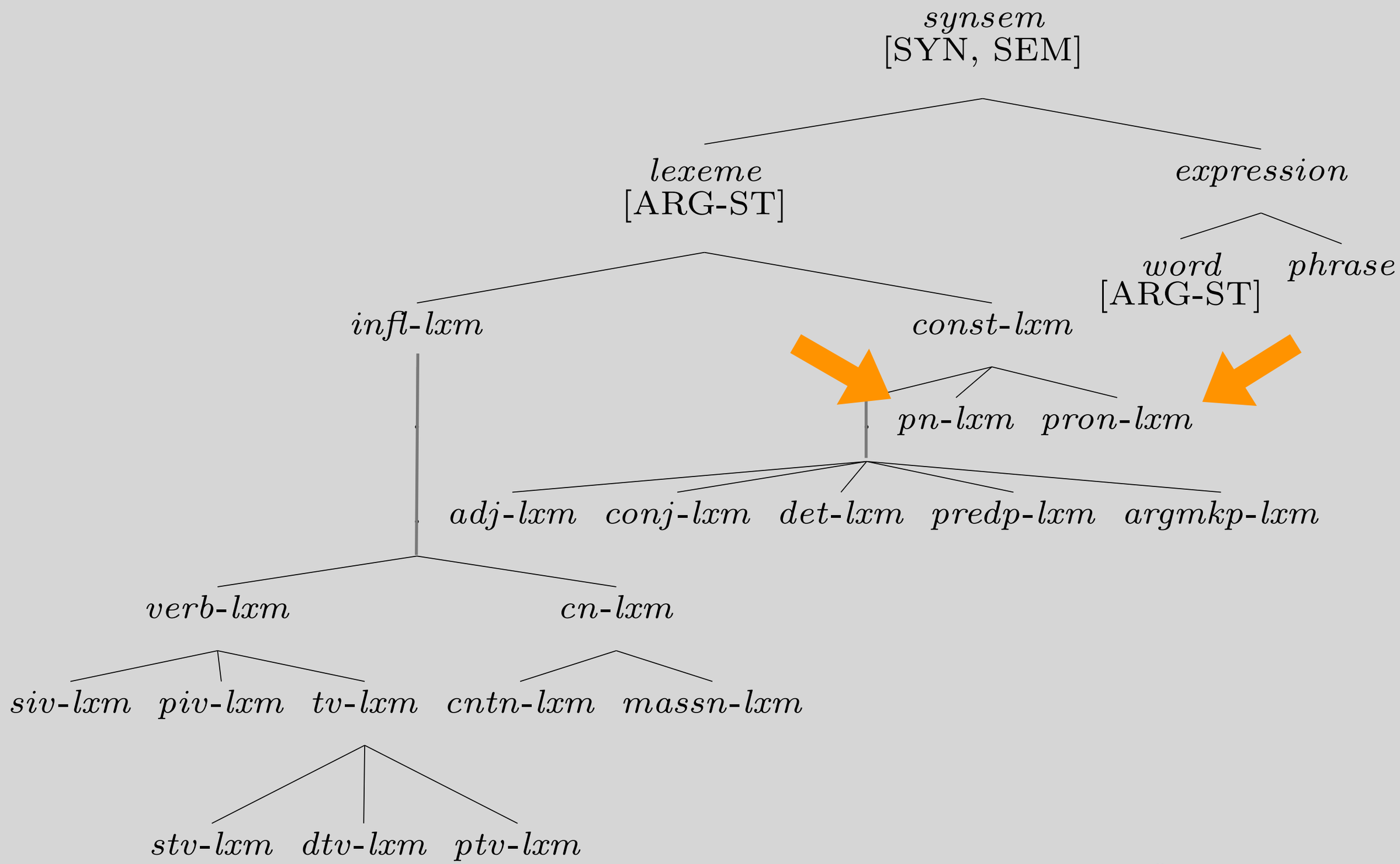
$$\textit{verb-lxm}\text{:}\quad \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \textit{verb} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{MODE} & \text{prop} \end{bmatrix} \\ \text{ARG-ST} & / \langle\ \text{NP},\ \ldots\ \rangle \end{bmatrix}$$

Subtypes of *verb-lxm*



- *verb-lxm*: [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*,...

Poll!

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

- Notation question. On p.237 we have example (24) which has COMPS /<> and (25) which has MOD /<>, which I understand to mean it's empty by default but can possible have arguments. In this case, are we using /<> instead of an optional flag like (NP) to keep the definition general? If so couldn't we have something like (X) as just an optional arg or list of arg in COMPS or MOD?

- Is it safe to say that “inviolable constraints” are all other constraints that are NOT inherited from the supertype? (because we are allowing these to override the default/ defensible ones, it feels a little wild to me to allow basically anything else to be inviolable)

Reading Questions

- Why do we have "/" representing two different things? In the textbook, it said that "/" can be used to indicate that a certain specification is defeasible but it can also be used to indicate that two features are identical by default. Why not have two different symbols?

- Can we override a defeasible constraint to make it underspecified? If so, how is this notated?

Reading Questions

- p. 243 states that when predicational prepositions appear as complements of verbs (as in I wrapped the blanket around me), non-empty MOD is irrelevant. Does this mean that we can just say [MOD <>] for the structure of 'around' in this particular sentence? Otherwise, should we say [MOD / <>] following the default from its lexeme?

Reading Questions

- When developing lexeme categories/ subcategories/subsubcategories/etc, what sorts of heuristics do linguistics use to draw the line between what warrants a new subcategory versus what are considered exceptions that should go onto lexical entries?

Reading Questions

- Why is the ARG-ST list written on its own level, outside of SYN or SEM? Does it fall into neither of those categories, or both?
- Is there a reason that in (32) on page 239 we can't have the same DP entry in both the SPR and ARG-ST Lists?
- In (30) on page 239, constraints are given for count nouns and mass nouns to take COUNT+ and COUNT- determiners. Is there a reason that these types of constraints are defined in ARG-ST rather than in SPR/COMPS? Is there a preference for one or the other?

Reading Questions

- On p. 242, why do we use <X, Y, Z> for the ARG-ST list on the lexical entry of give in (38), but we use <NP, NP, NP> as the ARG-ST list for the lexical sequence in (39)? Why not use <NP, NP, NP> for both?

Reading Questions

- To account for verbs which take PPs headed by certain prepositions, will we simply be able to add subtypes of piv-lxm and ptv-lxm that specify which preposition or set of prepositions they take in ARG-ST?
- How do optionally transitive verbs like “eat” fit into the new verb-lxm class? It doesn’t seem to fit into any of the subclasses - do we need a new subclass, or to make the complement on stv-lxm defeasible, or something else?

Reading Questions

- When does a family of lexical sequences become a lexeme? Do lexemes have to be "leaves" of the *-lxm subtypes?
- Why are there infinitely many lexical sequences that satisfy the lexical entry in (31)? And why does (32) represent a family of sequences?
- How would lexemes be reflected in parse trees (such as what we have done in HWs)?

Reading Questions

- What is the motivation for separating lexemes and words as separate entities. Is it that a lexical entry like (38) for *give* would encompass a family of lexical sequences which would include things like *give*, *gives*, and *given*? Or would *gives* and *given* still be separate from the lexical entry for *give*?

- What is the difference between inflection and lexical rules?

Reading Questions

- Is it possible to use lexeme constraints to constraint honorific speech patterns in languages such as Japanese, where three different types of honorific speech require different kinds of words?
- How does the lexeme paradigm with defeasible constraints generalize to other languages? Are there examples of this paradigm faltering?

Reading Questions

- Does the development of HPSG grammar in this book reflect the development of HPSG in the real world (for example, the introduction of Lexemes)? Or is this just for learning purposes?