Ling 566
Feb 4, 2019

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

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*,...

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?

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

- That is, default constraints are ‘pushed down’

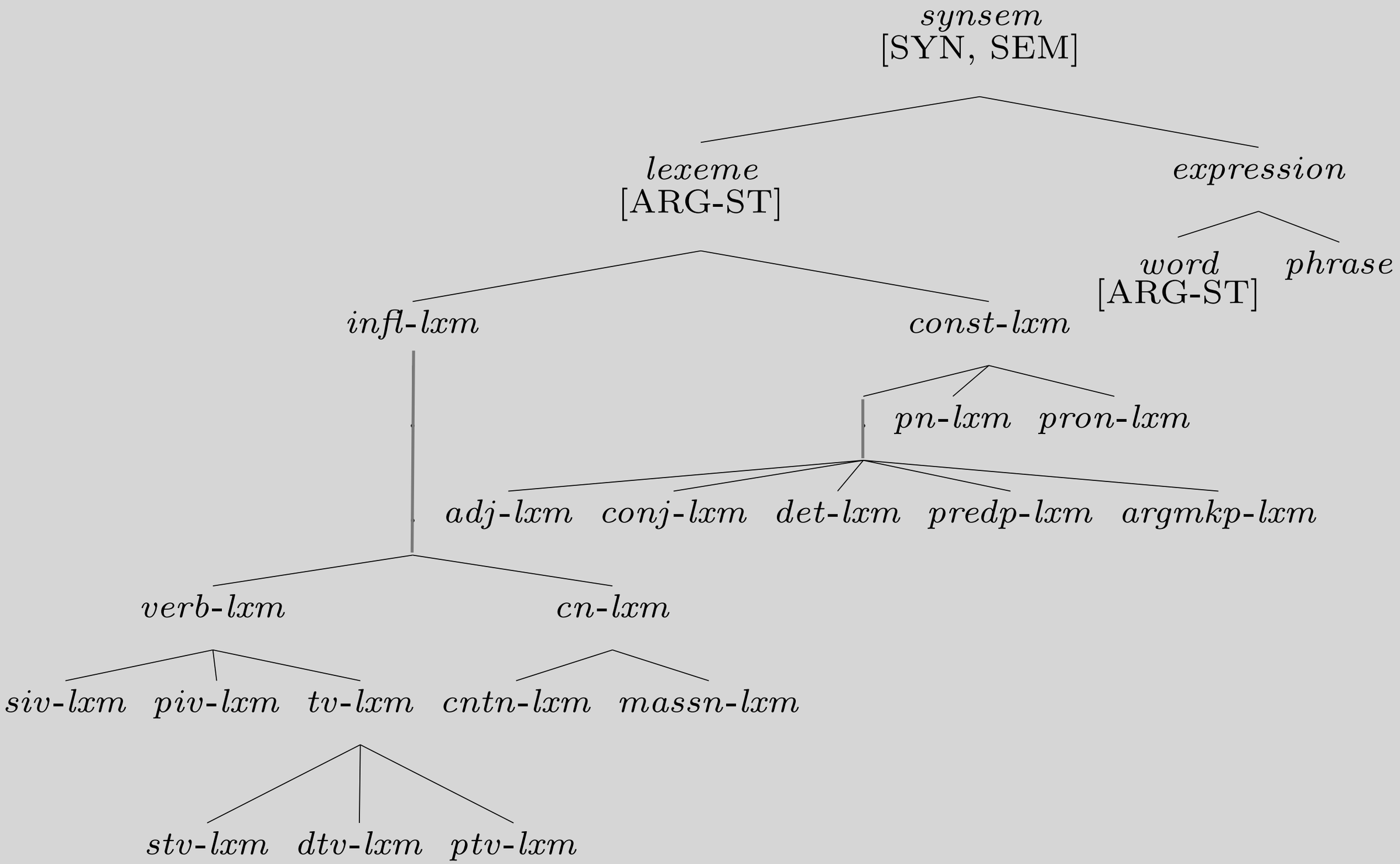
Question on Default Inheritance

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.

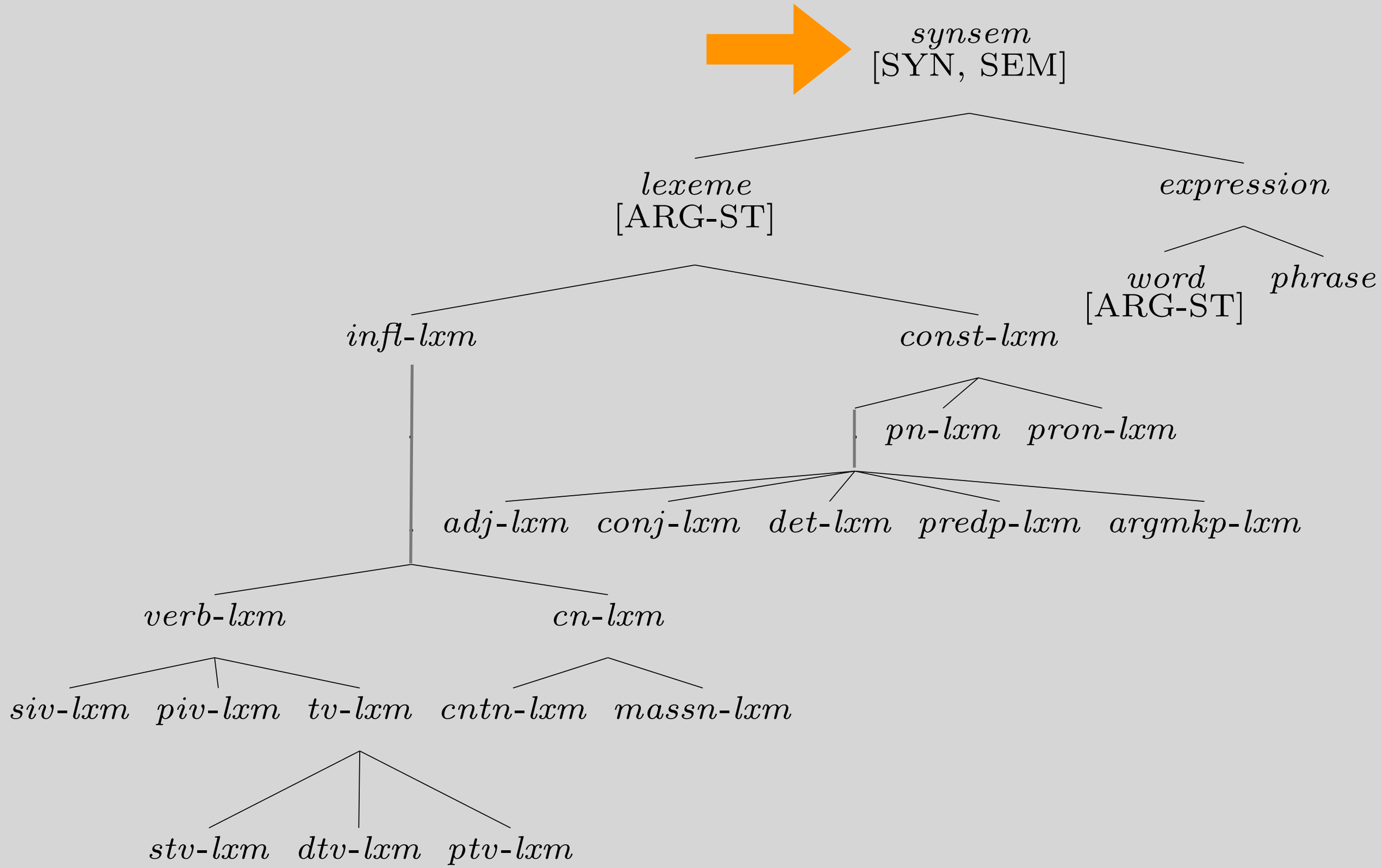
Our Lexeme Hierarchy



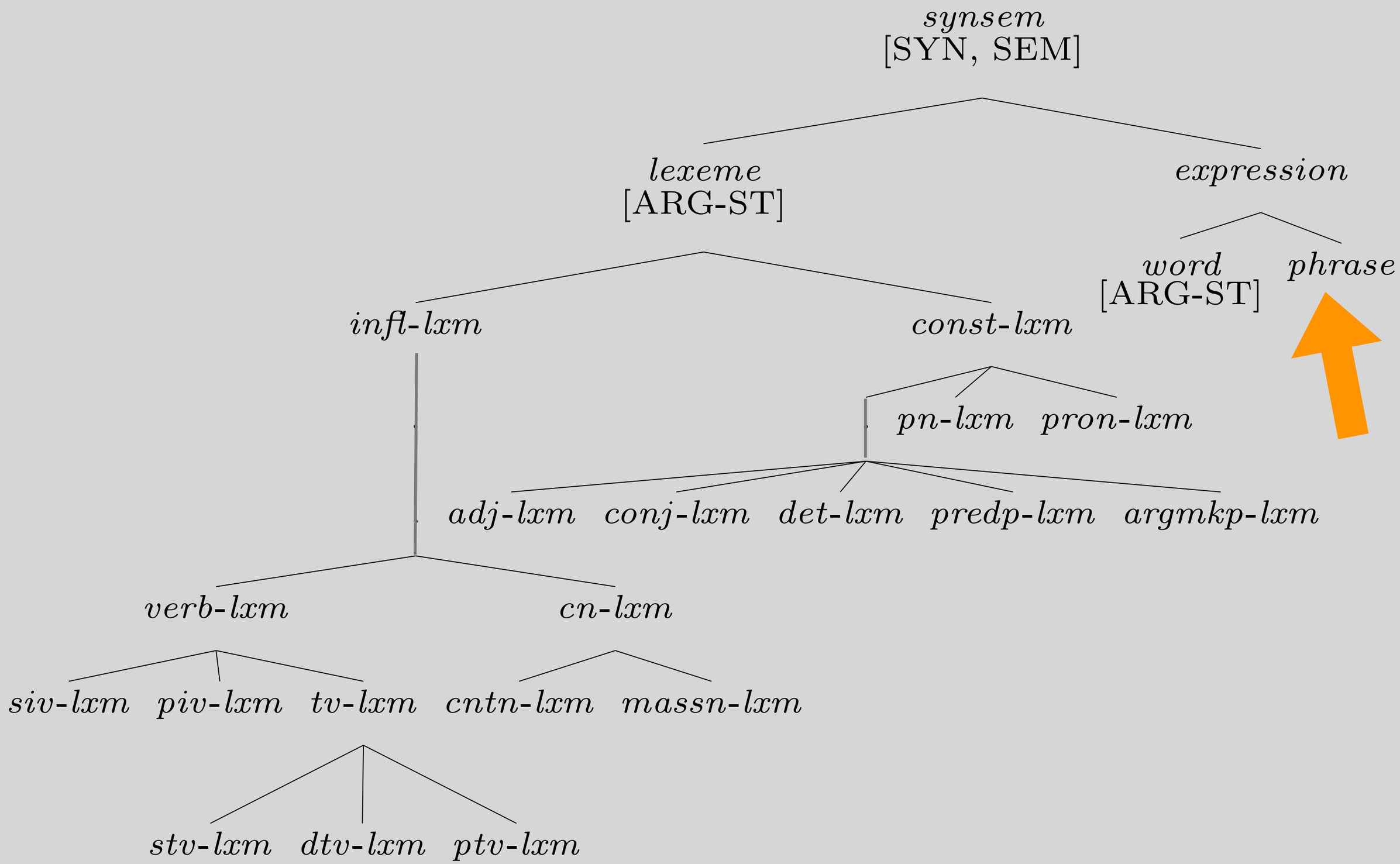
Functions of Types

- 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

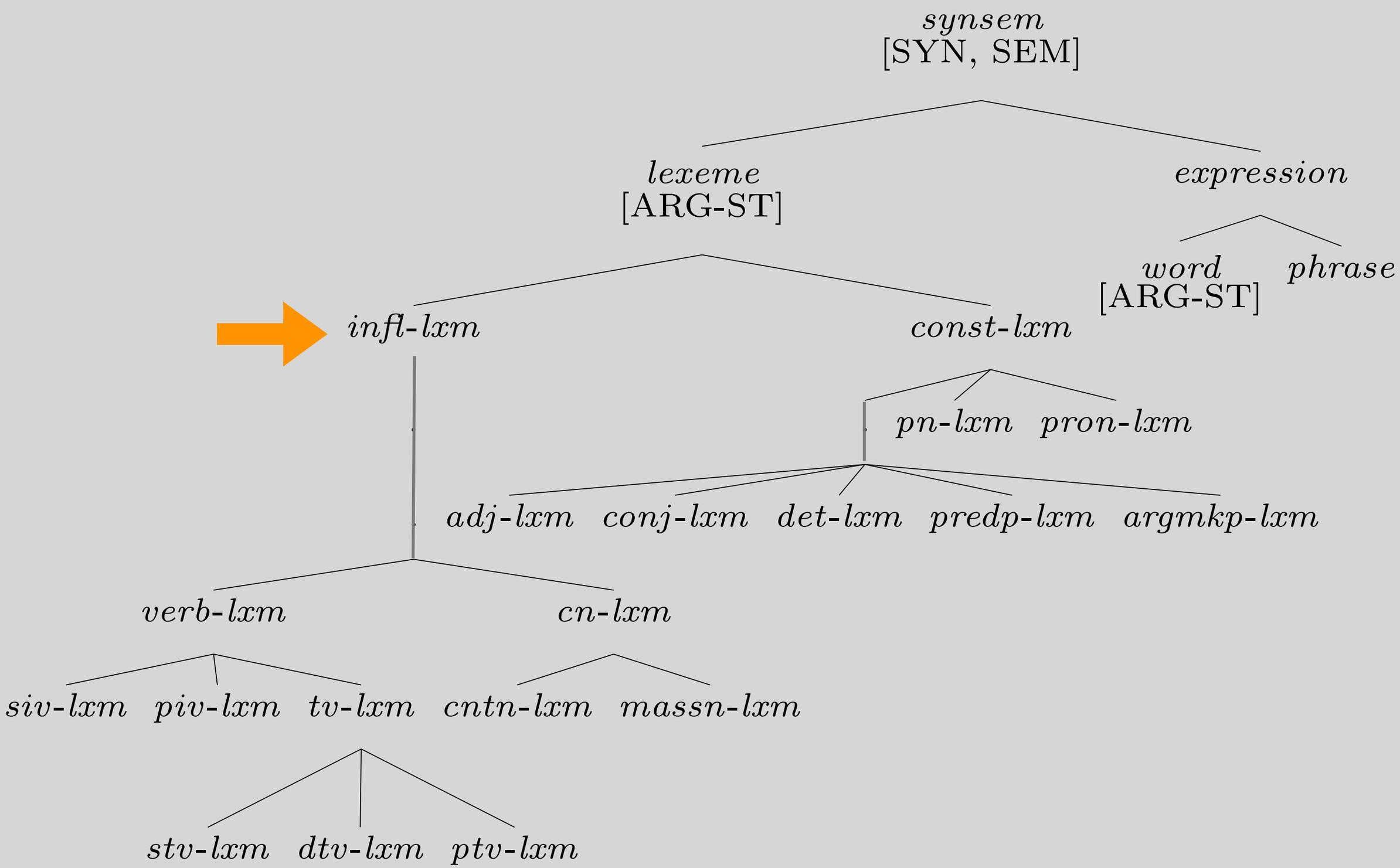
Every *synsem* has the features SYN and SEM



No ARG-ST on *phrase*



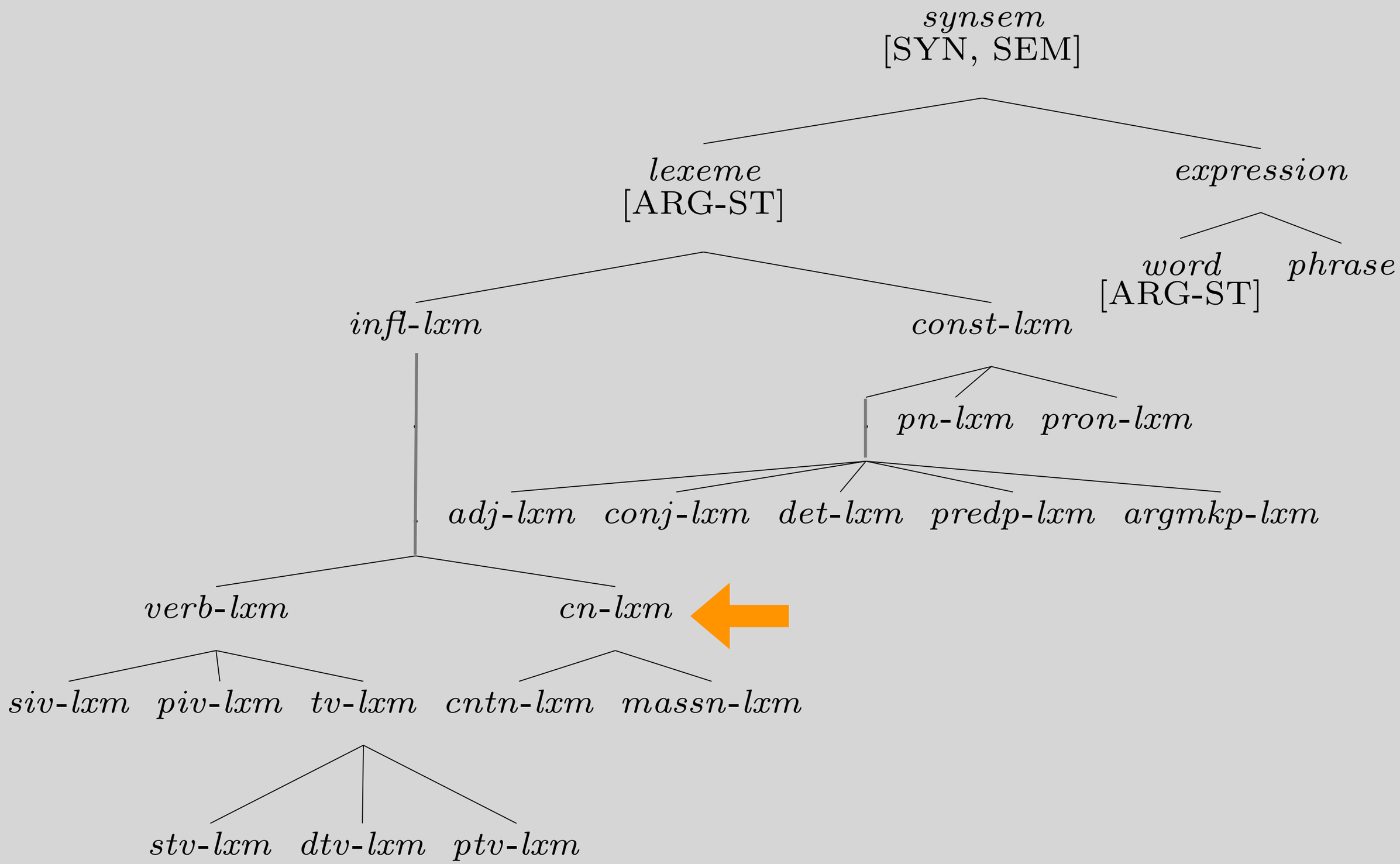
A Constraint on *infl-lxm*: the SHAC



A Constraint on *infl-lxm*: the SHAC

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

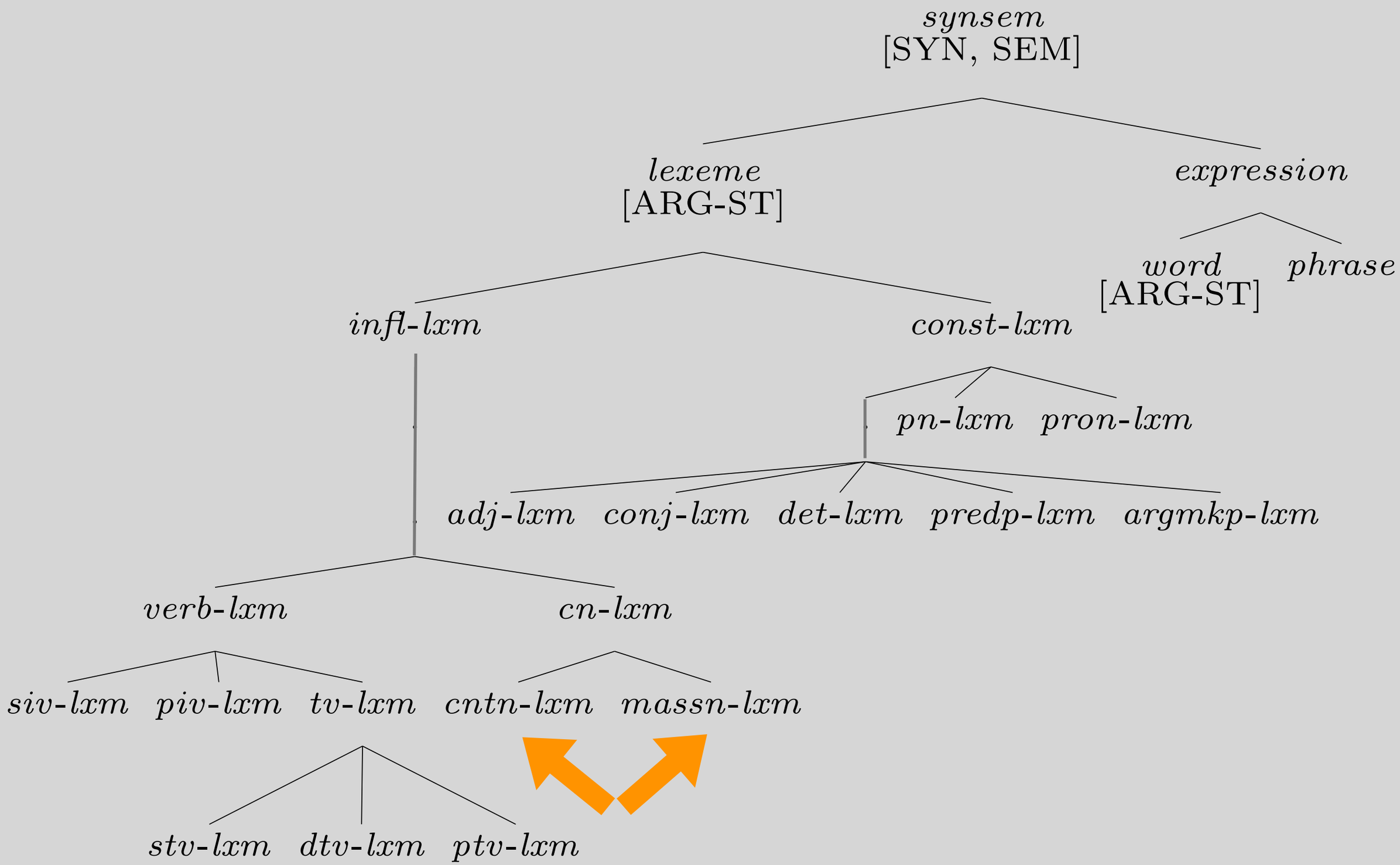
Constraints on *cn-lxm*



Constraints on *cn-lxm*

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

Formally Distinguishing Count vs. Mass Nouns

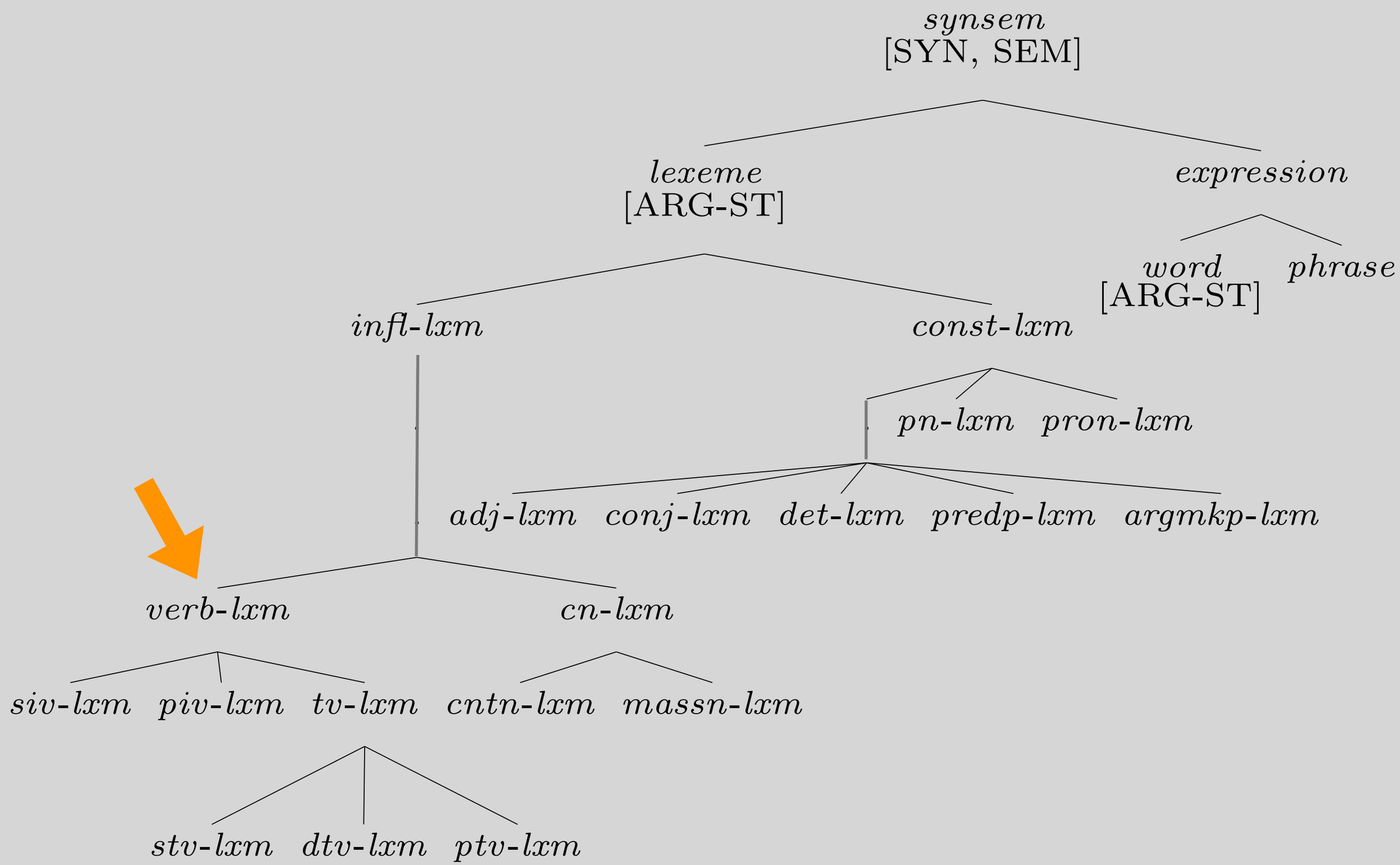


Formally Distinguishing Count vs. Mass Nouns

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

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

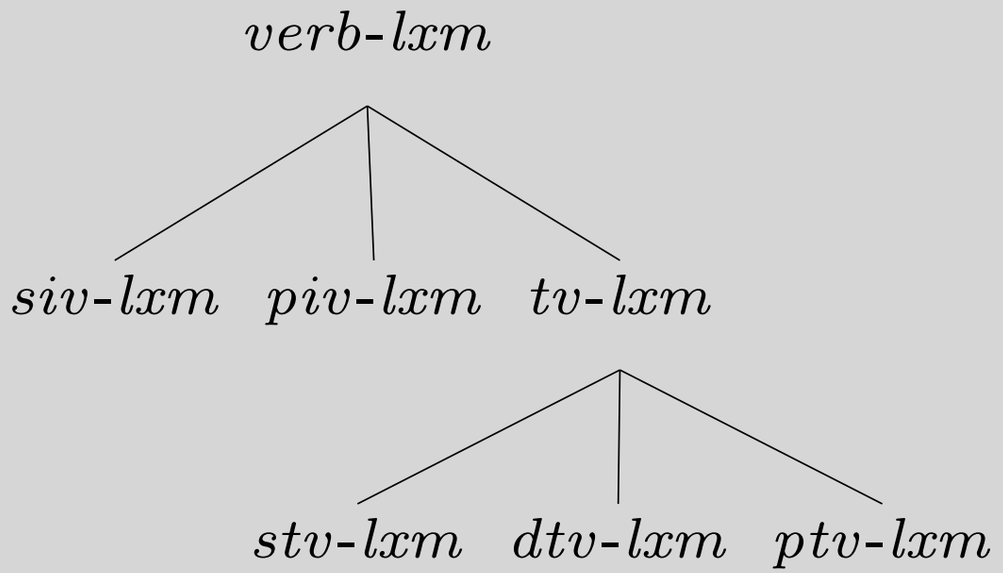
Constraints on *verb-lxm*



Constraints on *verb-lxm*

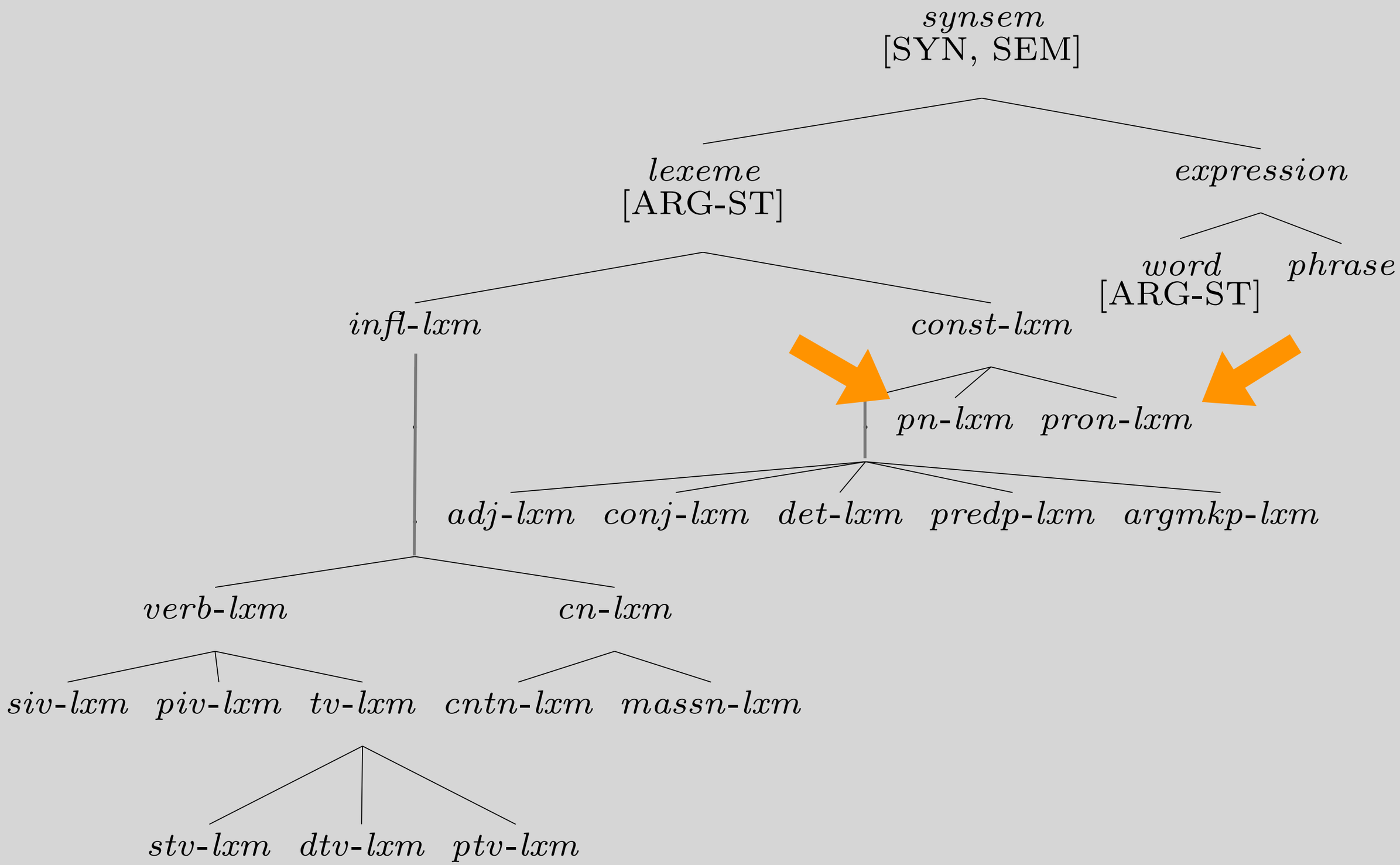
$$\textit{verb-lxm}: \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \textit{verb} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{MODE} & \text{prop} \end{bmatrix} \\ \text{ARG-ST} & / \langle\ \text{NP}, \ldots\ \rangle \end{bmatrix}$$

Subtypes of *verb-lxm*



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

Overview

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

Reading Questions

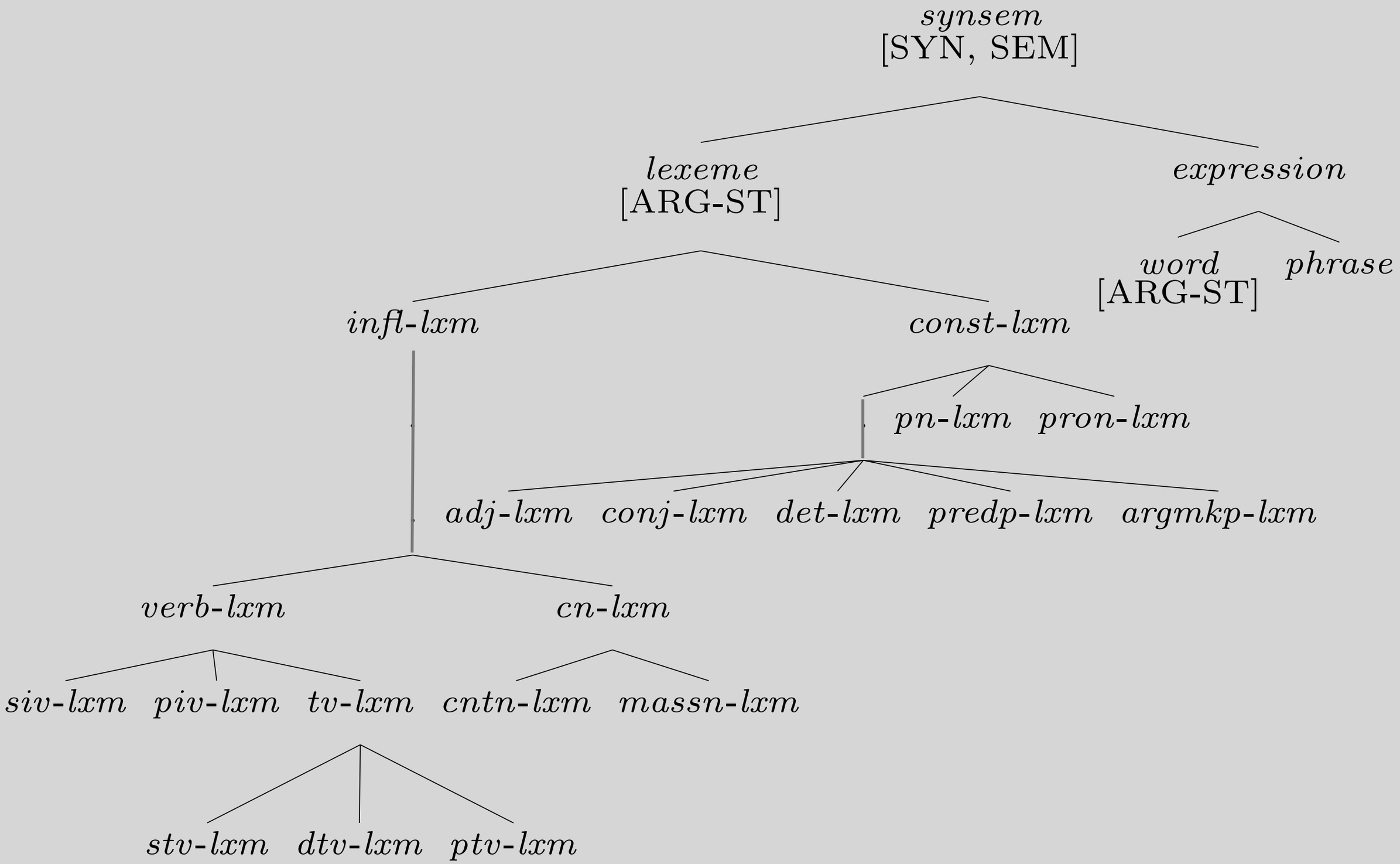
- "[the type hierarchy] allows us to stipulate common combinations of feature values only once, using (default) inheritance to account for their distribution." I thought this was already the de facto for most principles (e.g., the Valence Principle). How does a subtly distinct, lexeme-driven type hierarchy formalize this?

Reading Questions

- Is synsem in our reorganized type hierarchy on pg 229 capturing all of the information that was beneath *feat-struc*? I see the hierarchy is not at a final state now, but does *synsem* continue to stay in that position?

- Why isn't lexeme a subtype of expression / why are the two on the same level in the tree in (3). Expressions are made of words and phrases and lexemes seem to house the same properties. With the given *run* example, it makes sense that *run*, *was running*, and *will run* should all be under the same lexeme, even counting the auxiliary verbs around them.

Our Lexeme Hierarchy



Reading Questions

- In 8.3 pg 234 it says "Each (basic) lexical entry describes a distinct family of lexemes each of which is an instance of a maximal type T_m " and I'm confused why a lexical entry is describing a family of lexemes instead of an individual lexeme. The lexeme we develop in 8.4 seem specific enough that a lexical entry would correspond to just one or maybe I'm misunderstanding what is meant by "describes" here.

Reading Questions

- In the beginning of section 8.4, the textbook mentions that a certain kind of "lexical sequence" consisting of a "phonological form" and a "feature structure of type lexeme". Does this mean that phonology will become an important part of constructing a feature structure/lexical entry? Or does the phonological representation not really matter all that much?

Reading Questions

- I am confused about the X, Y first notation introduced in (35) on pg. 241. Could you explain what exactly is supposed to be going through my head when I read this notation? I don't recall it ever being formally laid out, so it seemed to come out of nowhere for me. I'm assuming it indicates variables of some sort, but why is it that we can't just "drag" these values "down" from the AGR-ST of the parent type? Is this notation only used for ARG-ST and the VAL features of lexemes?

Reading Questions

- Pages 233-234 talk about defeasible constraints (which can be overridden) and inviolable constraints (which cannot). I suppose it's possible for an inviolable constraint to become a defeasible one (or vice versa) as a language evolves over time. Are we concerned with this at all, or is our grammar a snapshot of a grammatical variant only as it exists at precisely this moment in time? Are there certain classes of constraints which are inherently inviolable and are resistant to defeasibility even as the language changes?

Reading Questions

- Footnote 7 on Page 234 in Section 8.3 talks about how final descriptions of a lexeme have no defeasible constraints and hence the hierarchy is replaceable with a more complicated one (where there are no defeasible constraints to begin with). I am having trouble in understanding how this comes about and could you give an example of two hierarchies equivalent in this manner?

Reading Questions

- I'm confused a little by the notation difference in lists between "..." and ⊕ /<>. Usually I take "..." to mean there may be more but we don't know what it is, but doesn't saying "plus an potentially empty list" achieve the same thing? I know we've used "..." before for verbs to indicate tense information that is beyond the scope of the current discussion, are there other things here that are similarly being left out of things like adjectives (45) and nouns (30)?

Reading Questions

- We seem to favor the use of ARG-ST a lot, which puzzles me. Why not separately specify SPR and COMPS? I don't see any advantage of using ARG-ST instead of SPR and COMPS, but one disadvantage is that the ARG-ST is ambiguous -- you cannot derive a unique (SPR, COMPS) tuple from an ARG-ST list.

Reading Questions

- In the beginning of section 8.4, the textbook mentions that a certain kind of "lexical sequence" consisting of a "phonological form" and a "feature structure of type lexeme". Does this mean that phonology will become an important part of constructing a feature structure/lexical entry? Or does the phonological representation not really matter all that much?

Reading Questions

- The VAL constraints for type predp-lxm given on page 243 indicate that a predicational preposition will always take a specifier and select something to modify. The text indicates that in the sentence "I wrapped the blanket around me," the NP "the blanket" is the specifier of around, however my first instinct would be that "the blanket" is what is being modified, and therefore should be indicated in the MOD list. Why isn't this the case? And furthermore if "the blanket" is the specifier, then what element of this sentence would be indicated in the preposition's MOD list?