Ling 566
Oct 24, 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

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*,...

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’

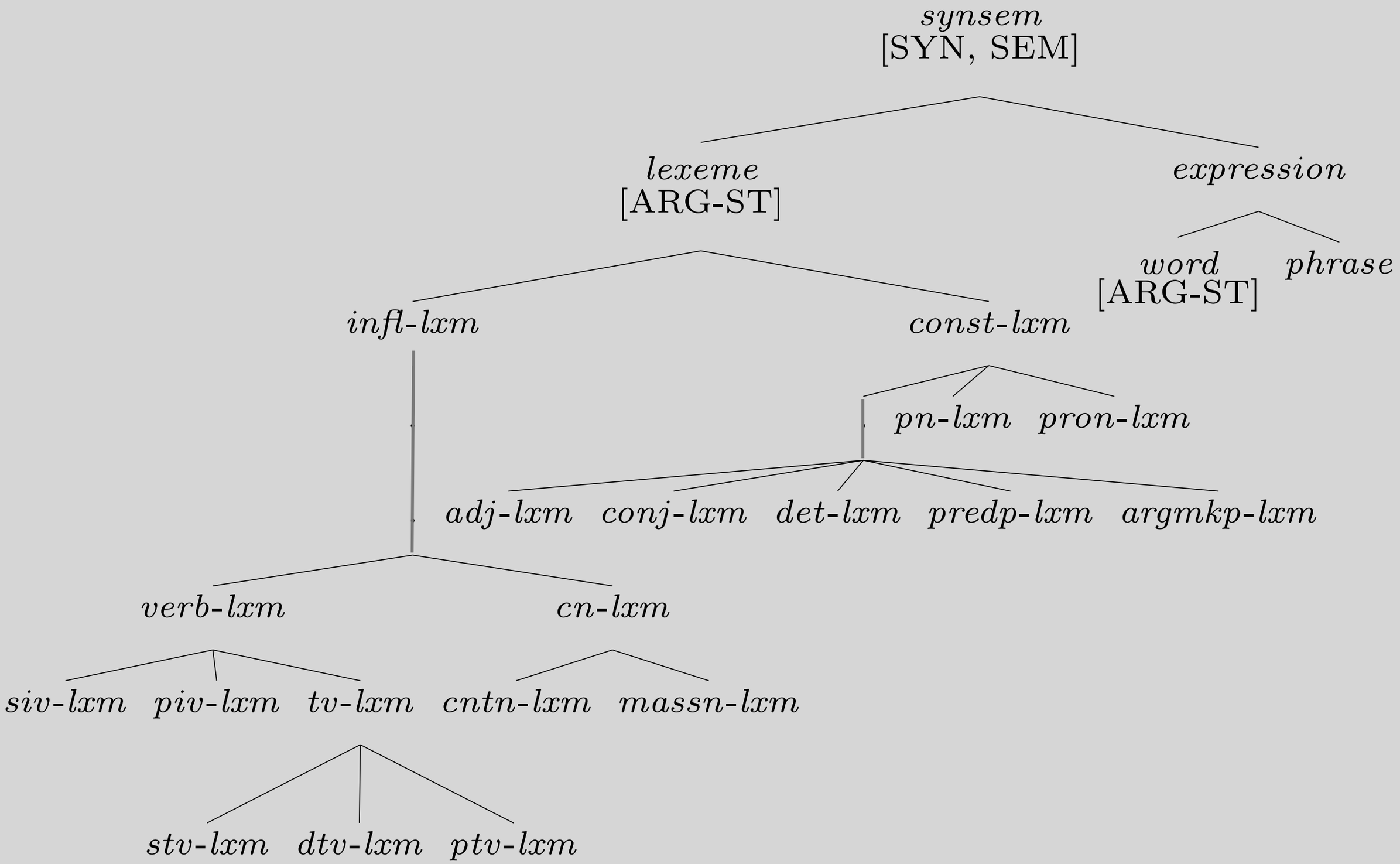
Question on Default Inheritance

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.

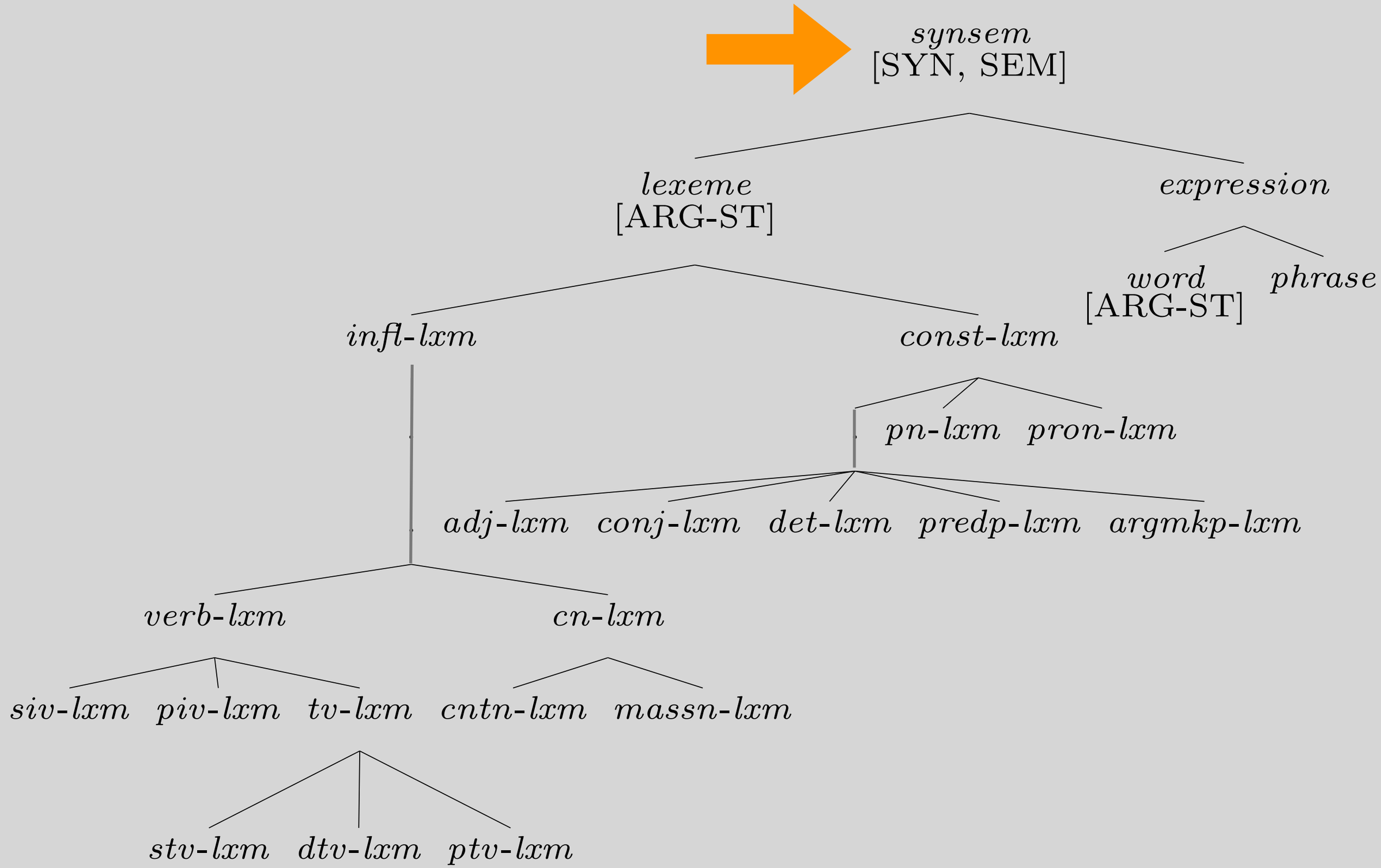
Our Lexeme Hierarchy



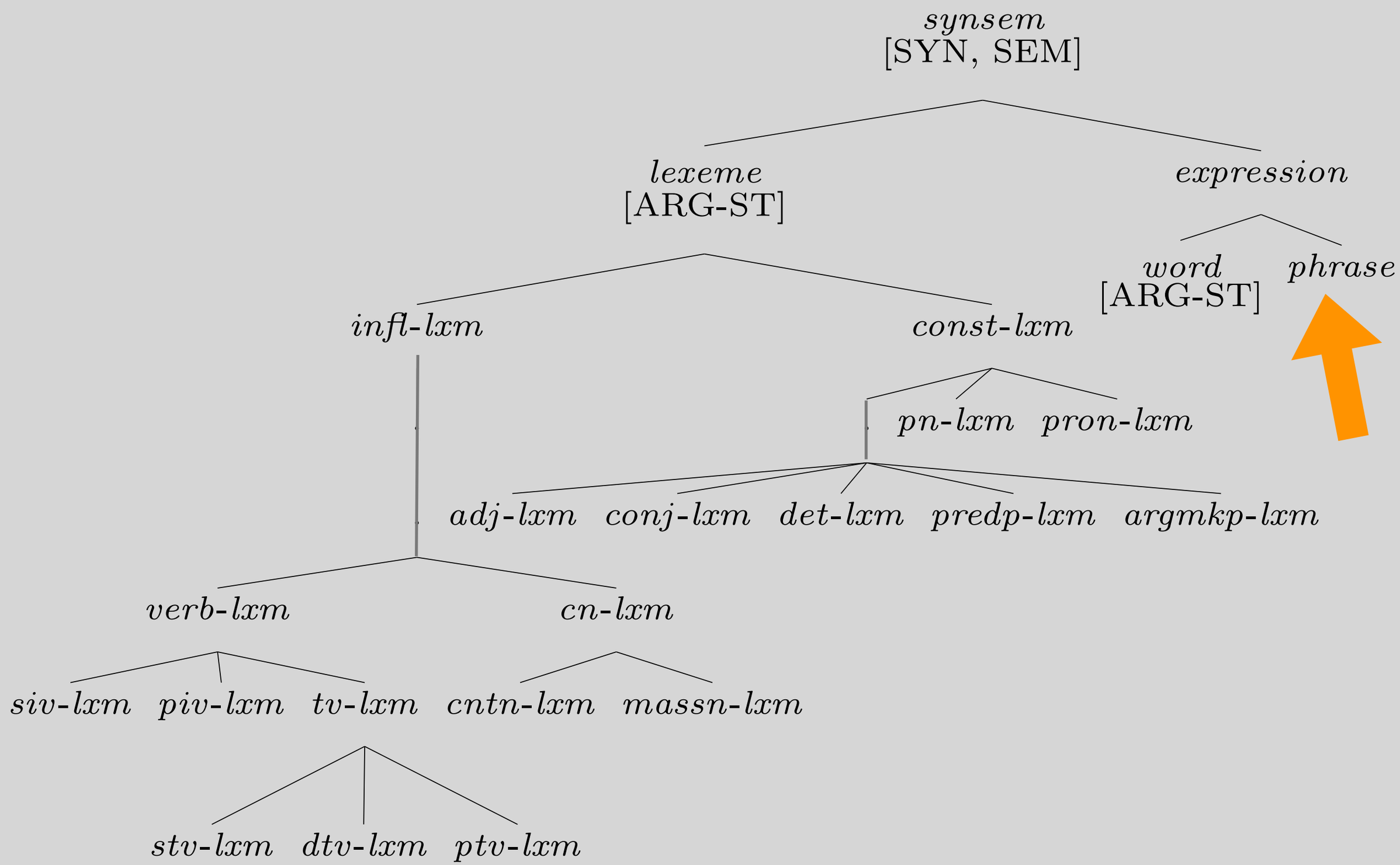
Functions of Types

- 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

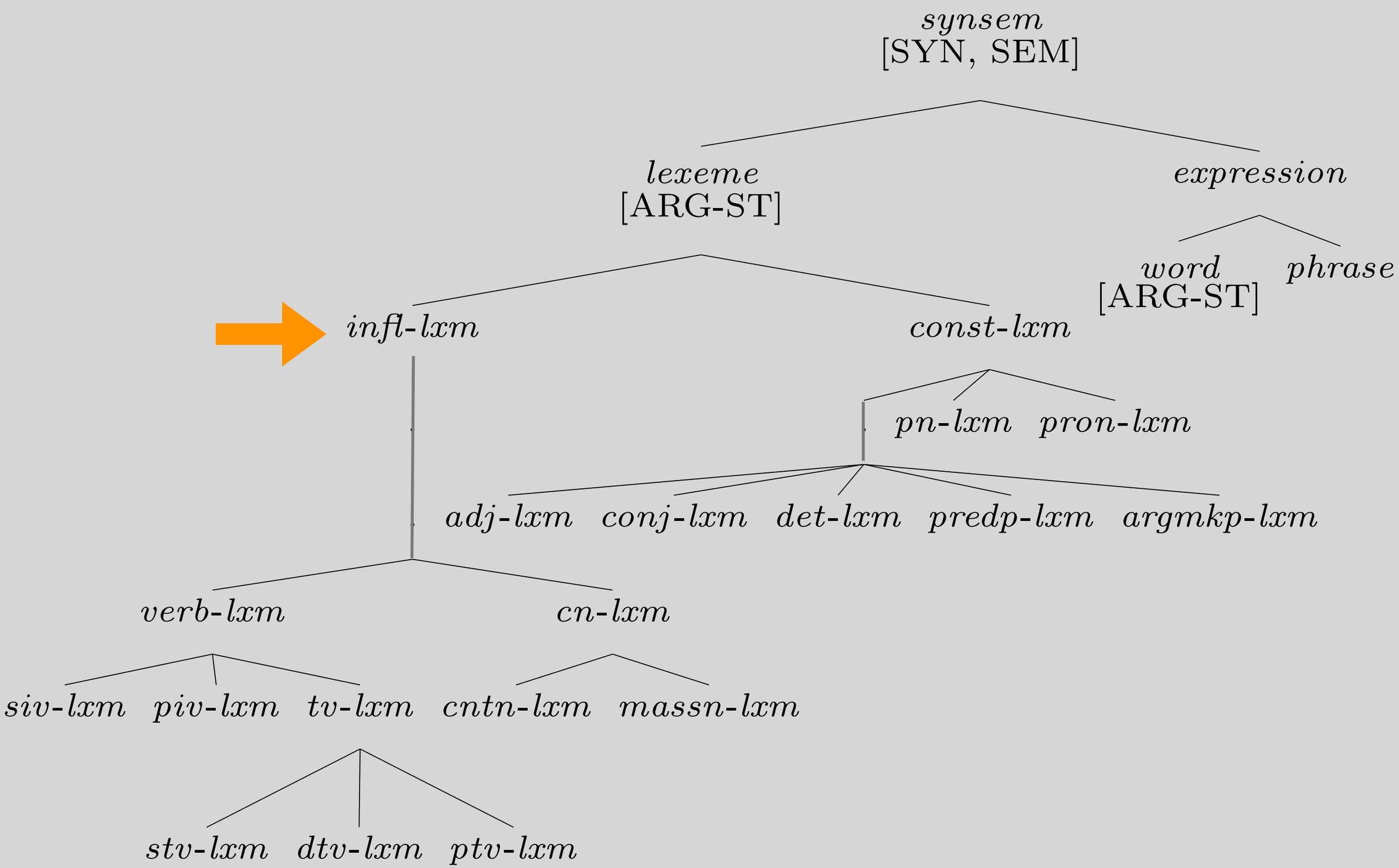
Every *synsem* has the features SYN and SEM



No ARG-ST on *phrase*



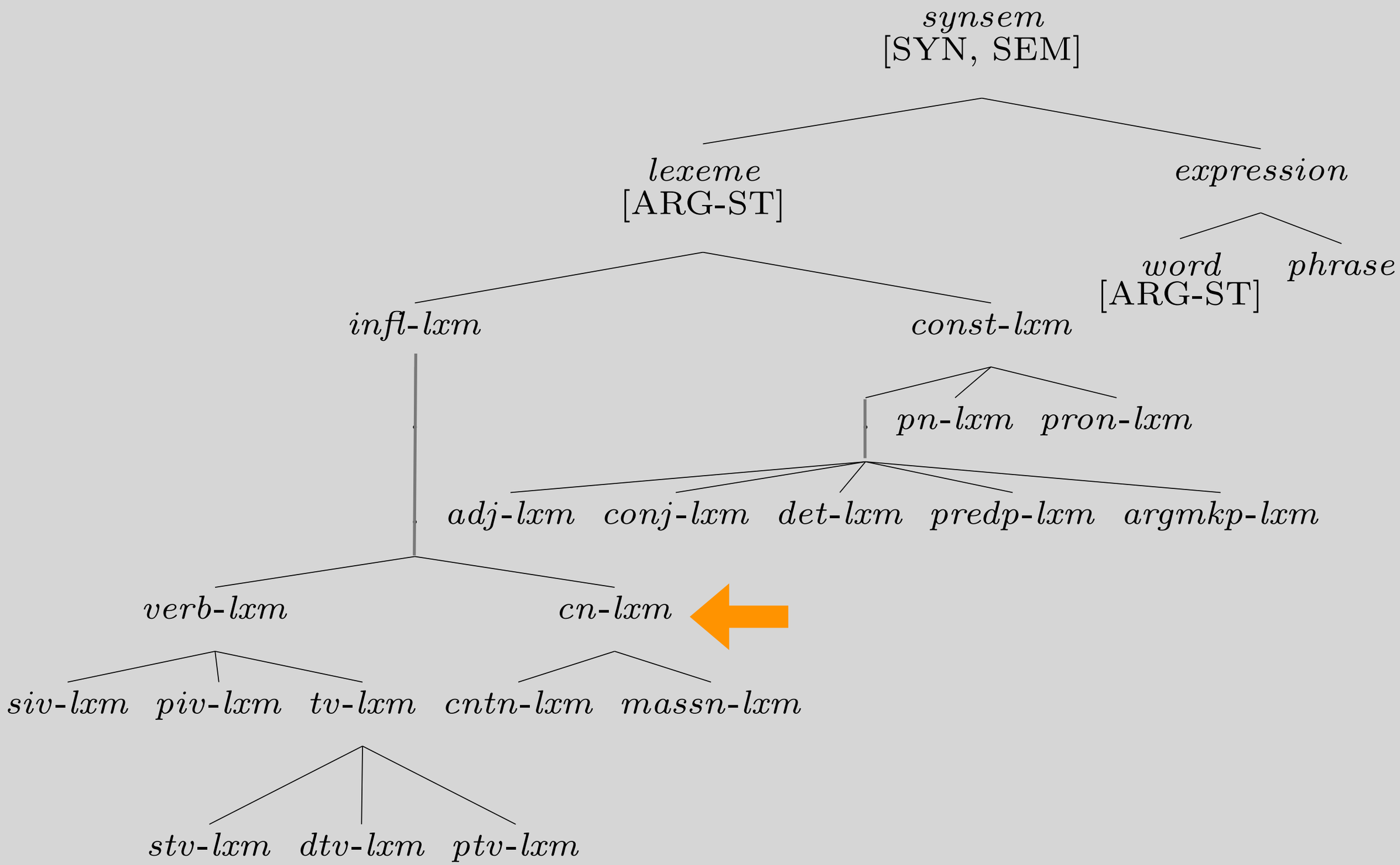
A Constraint on *infl-lxm*: the SHAC



A Constraint on *infl-lxm*: the SHAC

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

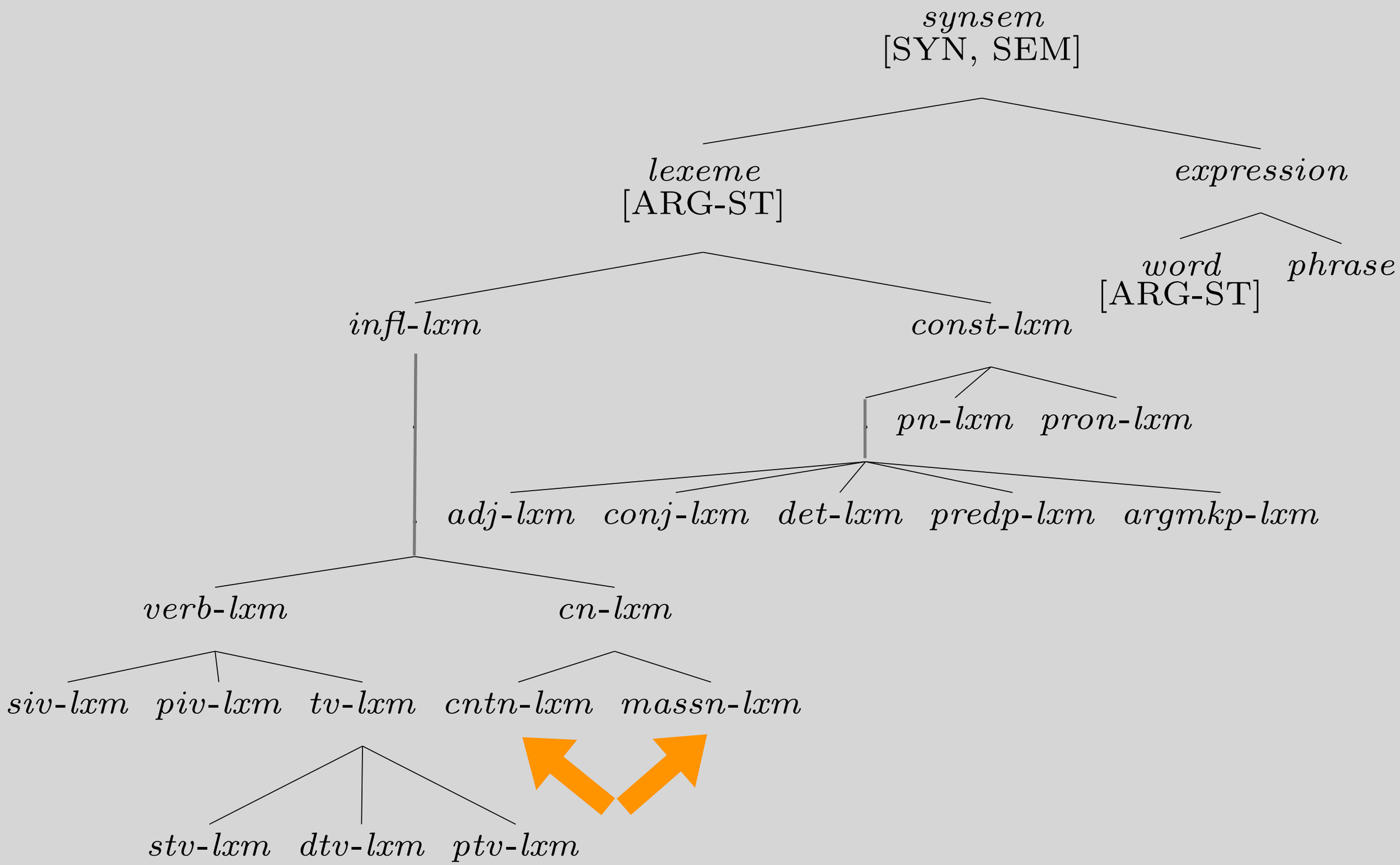
Constraints on *cn-lxm*



Constraints on *cn-lxm*

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

Formally Distinguishing Count vs. Mass Nouns

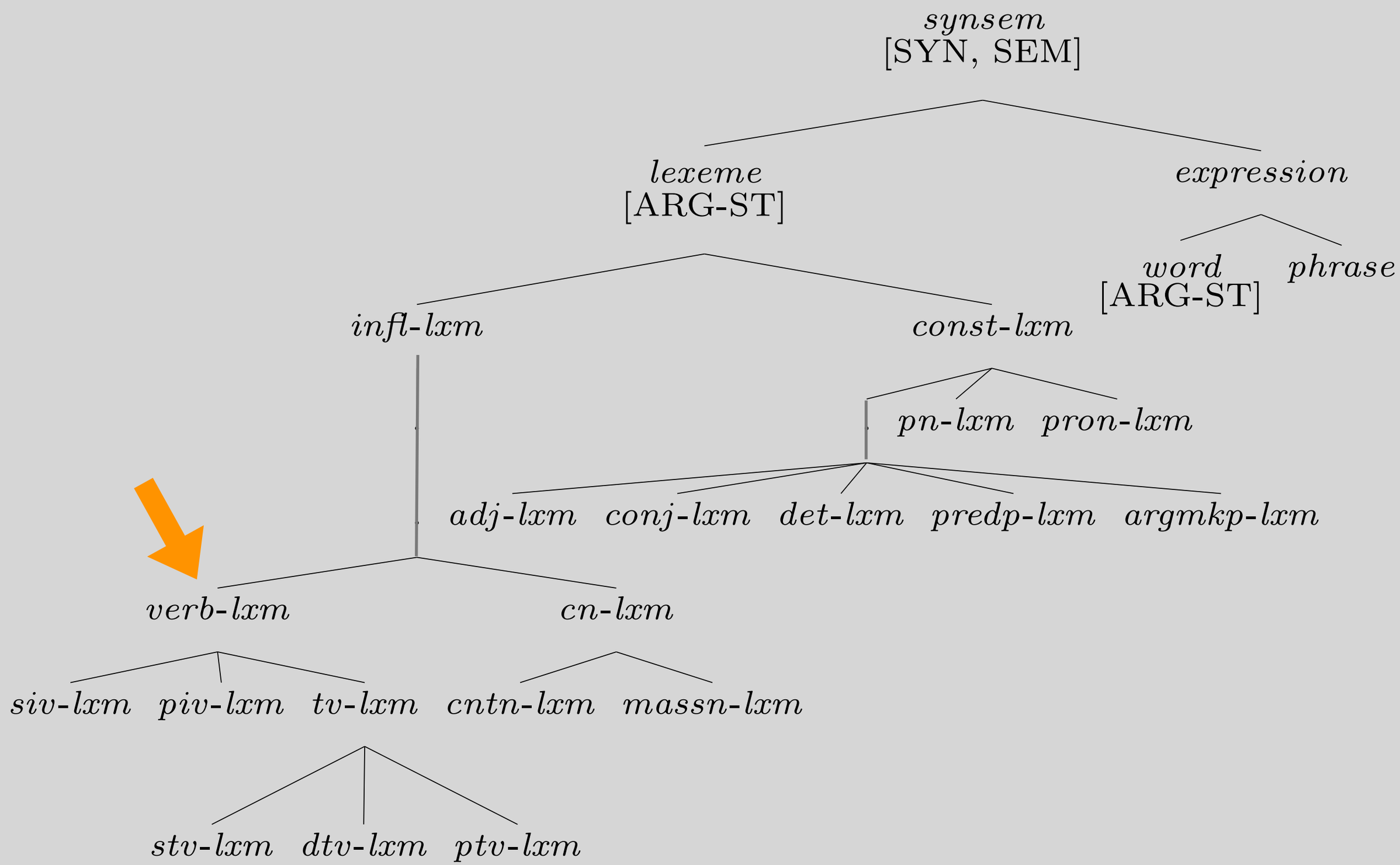


Formally Distinguishing Count vs. Mass Nouns

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

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

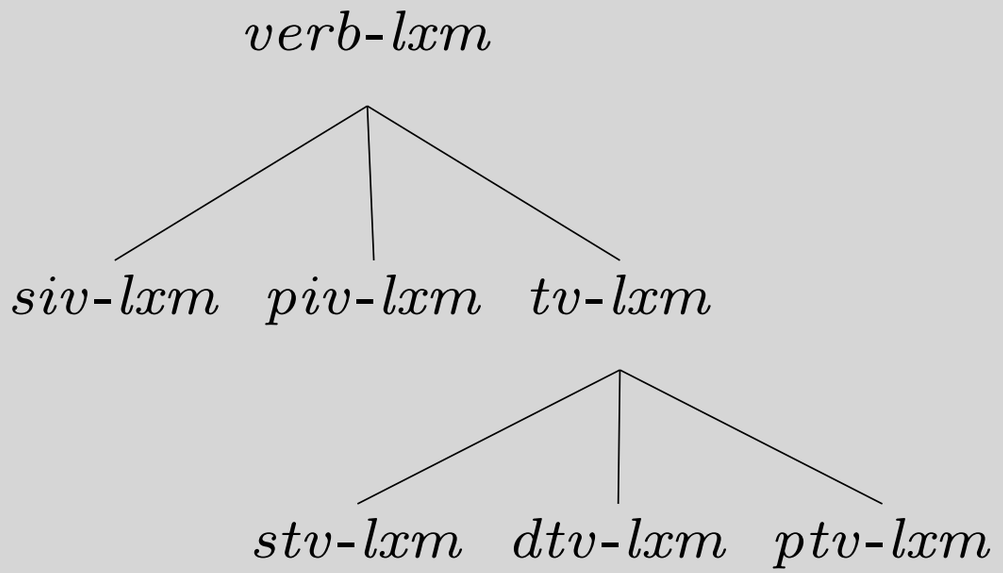
Constraints on *verb-lxm*



Constraints on *verb-lxm*

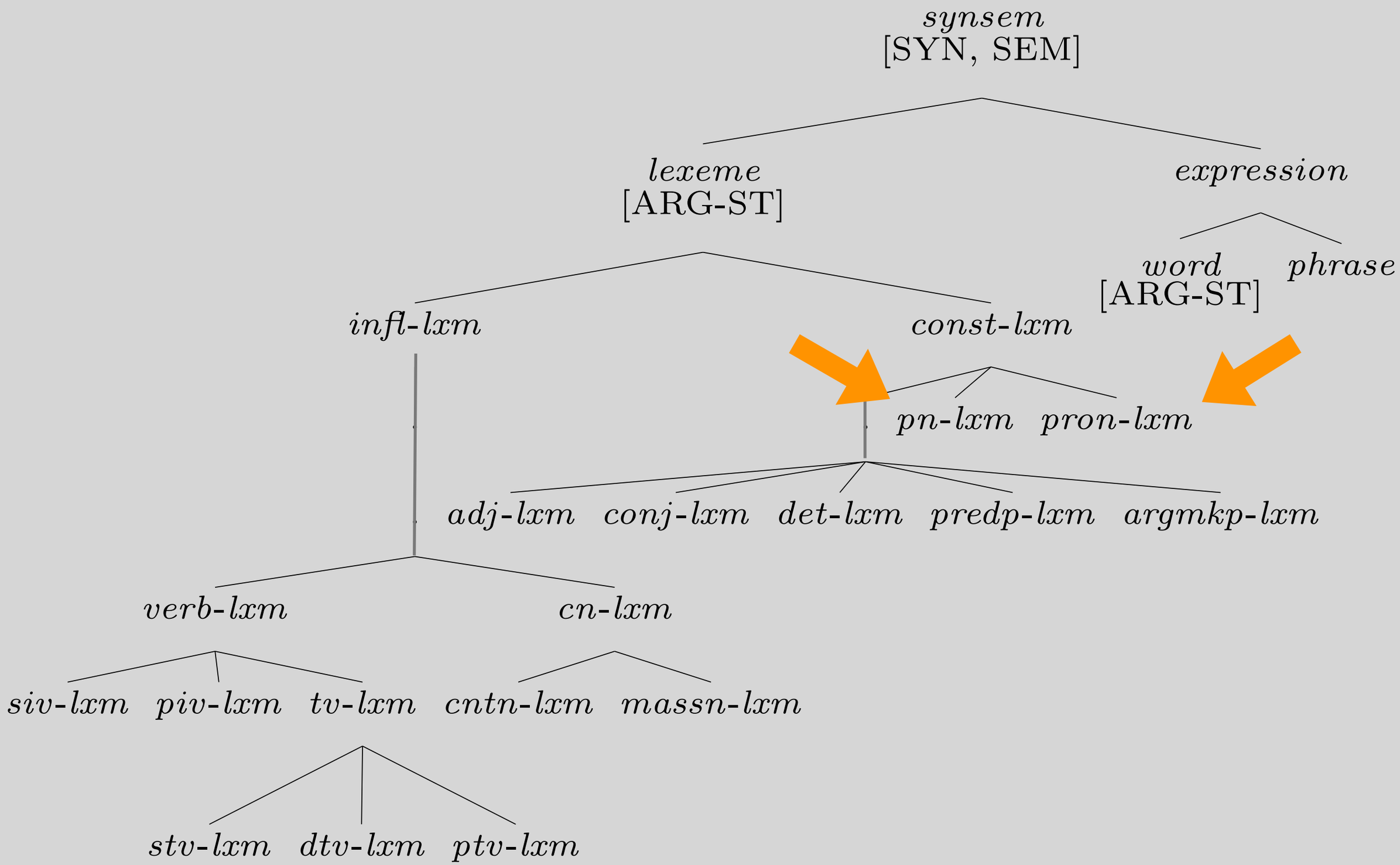
$$\textit{verb-lxm}: \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \textit{verb} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{MODE} & \text{prop} \end{bmatrix} \\ \text{ARG-ST} & / \langle\ \text{NP}, \ldots\ \rangle \end{bmatrix}$$

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

- Now that we are talking about scaling our lexicon what do we do for words that have multiple very different meanings. Take the word fly. Do we have one entry that represents the verb(act of flying like a bird), one the noun(the insect), and one that represents the adjective(one looking fly)?
- If there is a word with two different meanings, how would the lexeme be structured then? "His head was hurt" vs "He was headed to the mall" Does the lexeme take into account same word and different meaning?

Reading Questions

- What are some examples of lexemes that can be modifiers? It is stated in 8.4 that most cannot be modifiers.

Reading Questions

- For the compound nouns listed in (10) and (11), do they form only 1 lexical entry? If so, would that cause problems down the line?

Reading Questions

- A notation question, on pages 241-242, what do the letters such as X, Y and Z represent? I'm lost as to why some of the elements of the ARG-ST are these seemingly arbitrary variables, but some have a more familiar definition like an NP or PP.

Reading Questions

- Though there is an explanation in 8.4.4, I still do not entirely understand why there needs to be a distinction between lexemes and part of speech. Part of speech does govern which features are appropriate for a particular structure, but lexemes appear to do the same thing, albeit with certain requirements for feature values.

Reading Questions

- Are there ever conditions for the overridable rules described in this chapter? For example, if you had a variable x could you allow it to be overridden by some other variable y, but not any other variable?
- How do we know which rules are nondefeasible and which ones are defeasible? Is this information provided by the lexicon? Or the rules themselves?
- Are defeasible constraints similar to specifying a default value for objects of a given type X in OOP which can be overridden when creating a new object of type X?

Reading Questions

- "Defeasible constraints [are] constraints on a given type that hold by default, unless contradicted by some other constraint that holds at a more specific level". So are defeasible constraints only at one level of the hierarchy, whereas monotonic constraints pass down to their subtypes? The wording of this definition is confusing to me.

Reading Questions

- in examples (24) on page 236, where does the / verb come from?
- Ti does have SYN/[...], but Tj doesn't have /verb.

(22)

$$\mathrm{T}_i : \left[\begin{array}{l}\mathrm{SYN}\ \left[\begin{array}{l}\mathrm{VAL}\ \left[\begin{array}{l}\mathrm{MOD}\ \left\langle\left[\begin{array}{l}\mathrm{SYN}\ / \left[\begin{array}{ll}\mathrm{HEAD} & \textit{verb} \\ \mathrm{VAL} & \left[\begin{array}{ll}\mathrm{SPR} & \langle\ \rangle \\ \mathrm{COMPS} & \langle\ \rangle\end{array}\right]\end{array}\right]\end{array}\right]\right\rangle\end{array}\right]\end{array}\right]\end{array}\right]$$

(23)

$$\mathrm{T}_j : \left[\begin{array}{l}\mathrm{SYN}\ \left[\begin{array}{l}\mathrm{VAL}\ \left[\begin{array}{l}\mathrm{MOD}\ \left\langle\left[\mathrm{SYN}\ \left[\mathrm{VAL}\ \left[\mathrm{SPR}\ \langle\ \mathrm{NP}\ \rangle\ \right]\right]\right]\right\rangle\end{array}\right]\end{array}\right]\end{array}\right]$$

(24)

$$\mathrm{T}_i\ \&\ \mathrm{T}_j : \left[\begin{array}{l}\mathrm{SYN}\ \left[\begin{array}{l}\mathrm{VAL}\ \left[\begin{array}{l}\mathrm{MOD}\ \left\langle\left[\begin{array}{l}\mathrm{SYN}\ \left[\begin{array}{ll}\mathrm{HEAD} & / \textit{verb} \\ \mathrm{VAL} & \left[\begin{array}{ll}\mathrm{SPR} & \langle\ \mathrm{NP}\ \rangle \\ \mathrm{COMPS} & / \langle\ \rangle\end{array}\right]\end{array}\right]\end{array}\right]\right\rangle\end{array}\right]\end{array}\right]\end{array}\right]$$

Reading Questions

- Footnote 7 states that "[f]inal lexical descriptions of a lexeme or word contain no defeasible constraints" (pg 234). Is this because unless they are overwritten, then we can assume that defeasible constraints are present/hold and so it would be redundant to include them in the final description? Or is this getting at something else?

Reading Questions

- Looking at the general lexeme in (25) with a default constraint of having no modifier, I wonder what sorts of advantages there are to generalizing constraints? Conversely, should we be concerned if a defeasible constraint is often overriden?

- How is a defeasible list, especially a defeasible empty list, different from a list where items are optional (denoted by parentheses) or "choose either one" (denoted by vertical bar)?

Reading Questions

- How do we tell the different argument marking Ps apart, and get the right ones in the right place?

Reading Questions

- We had <ate, [COMPS <(NP)>]>.
- Now that *stv-lxm* and *siv-lxm* are different types, would we need two seperate lexical entries for ate, one [COMPS <NP>] and one [COMPS <>], derived from two lexemes? As we did with Ps that can be derived from both *argmkp-lxm* and *predp-lxm*? If so, are we eliminating optionality in COMPS list? Or do we have some sort of lexical rules for such verbs later?

Reading Questions

- Could you clarify the difference between a "lexical sequence" and "lexical entry"? I don't understand the distinction of lexical entries are descriptions/lexical sequences are models. Are lexical entries as a concept a subset of lexical sequences?

- Pg. 236: "Now that we have introduced the term 'lexical sequence', we will reserve the term 'lexical entry' for the pairings of form and linguistic constraints that we list in the lexicon. Lexical entries, like other parts of the grammar, are descriptions. Lexical sequences (both those that satisfy lexical entries and those licensed by lexical rules) are models."

Reading Questions

- "... each (basic) lexical entry describes a distinct family of lexemes, each of which is an instance of a maximal type Tm." Is Tm the lexeme in this hierarchy? If so, why doesn't each lexical entry in the hierarchy inherit from Tm , as opposed to Tm inheriting from the lexical entries, which is shown in structure (16) where Tm is a leaf node?" (p.234)