Ling 566
Oct 21, 2014

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 ARG-ST 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'

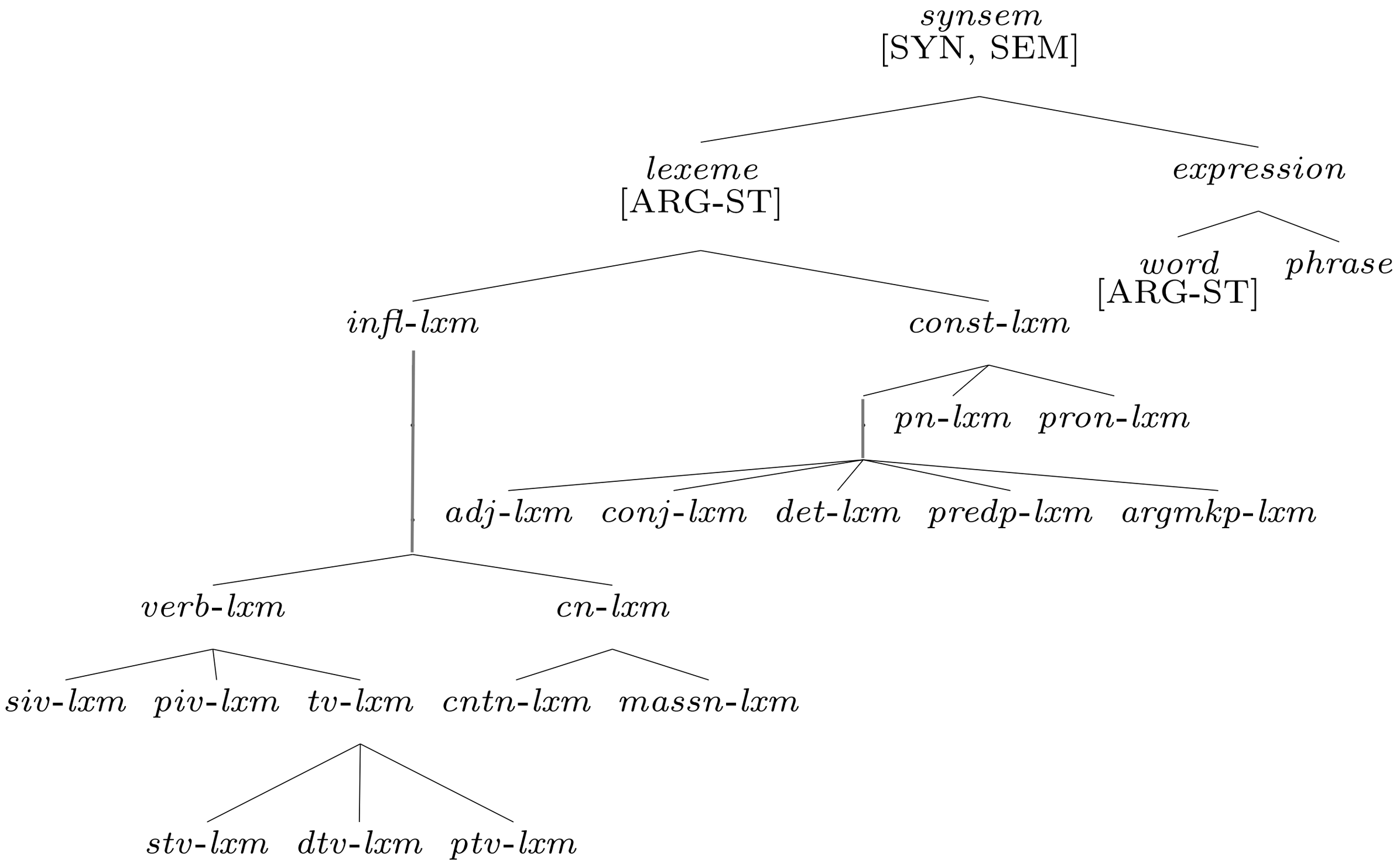
Question on Default Inheritance

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.

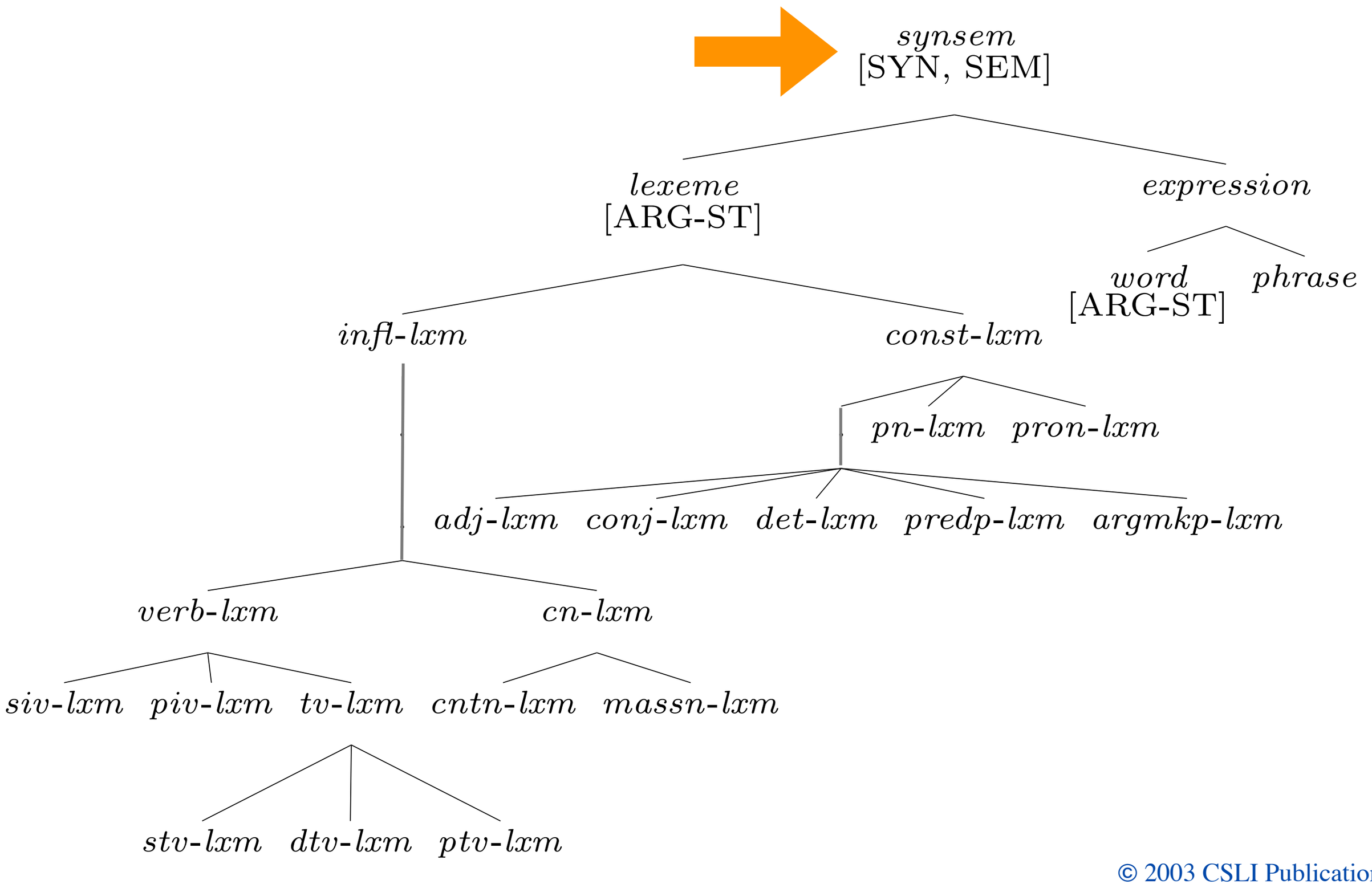
Our Lexeme Hierarchy



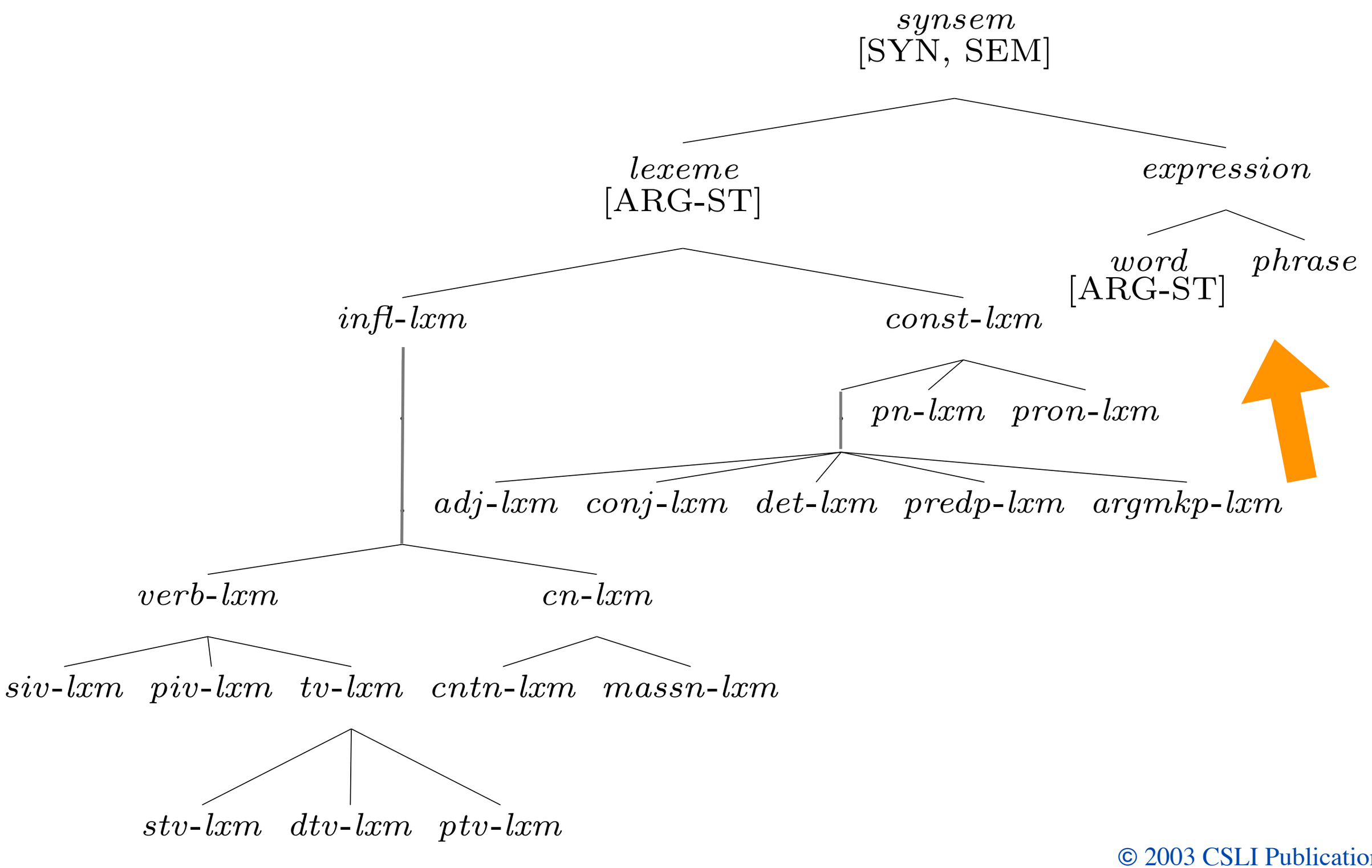
Functions of Types

- 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

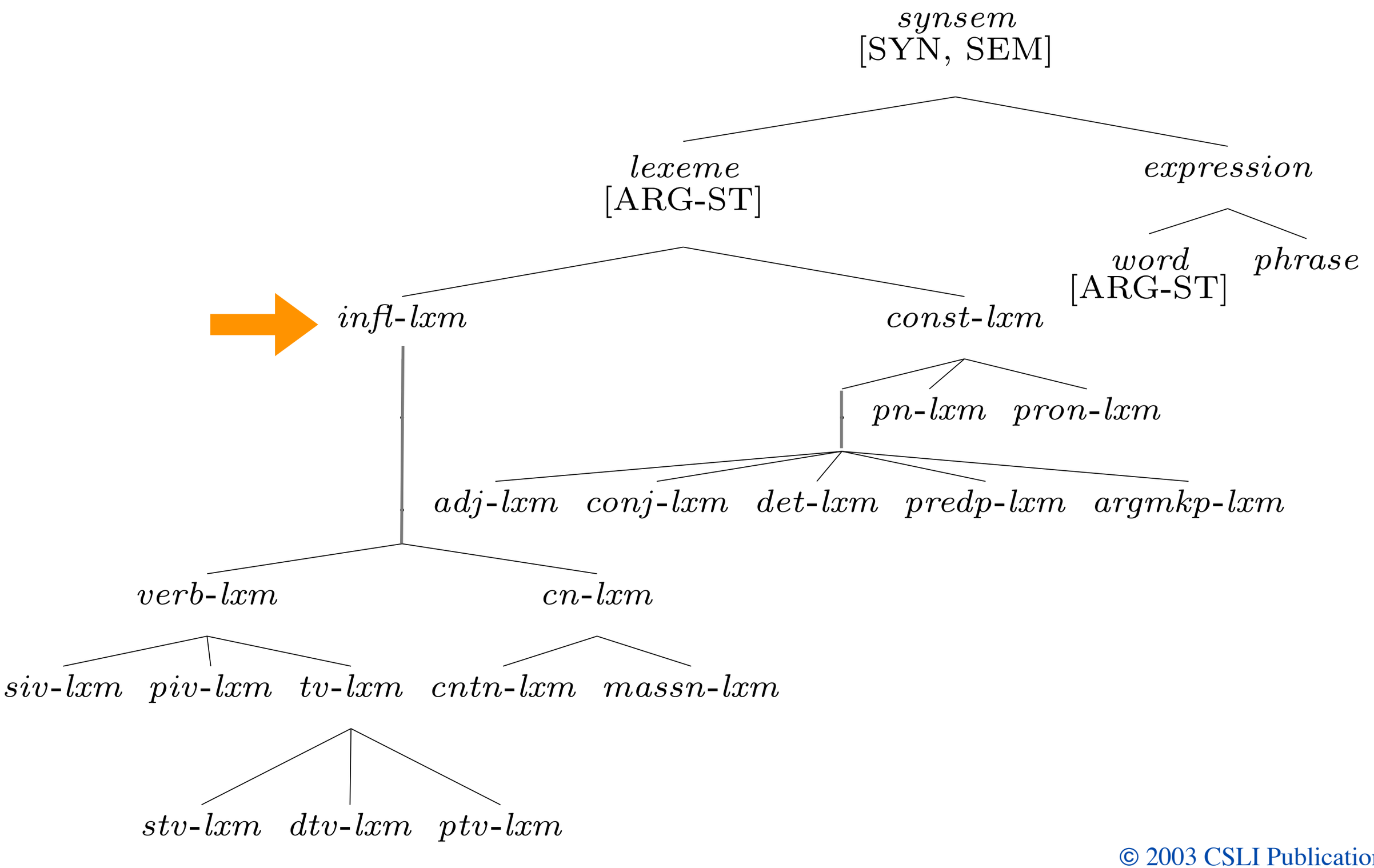
Every *synsem* has the features SYN and SEM



No ARG-ST on *phrase*



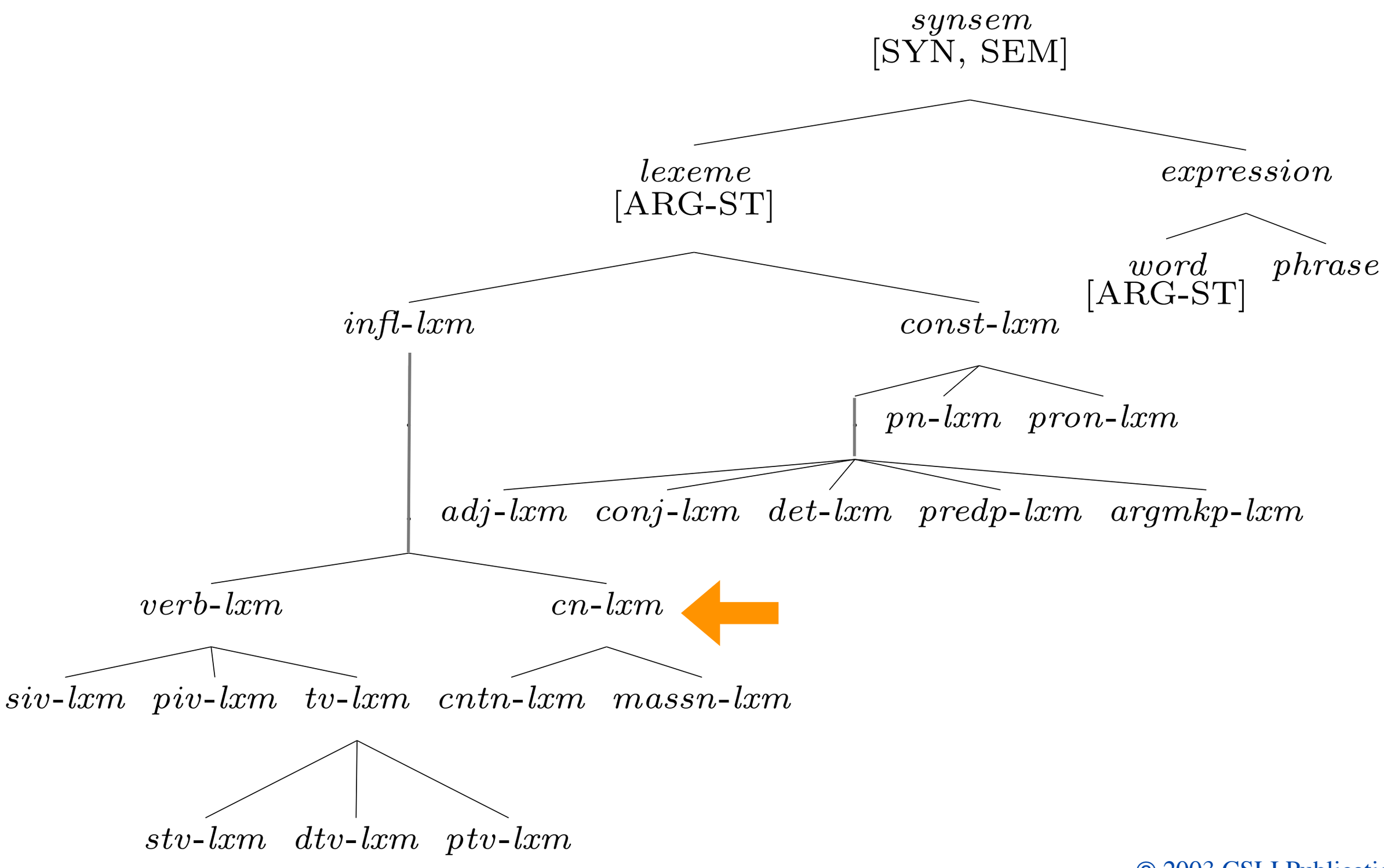
A Constraint on *infl-lxm*: the SHAC



A Constraint on *infl-lxm*: the SHAC

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

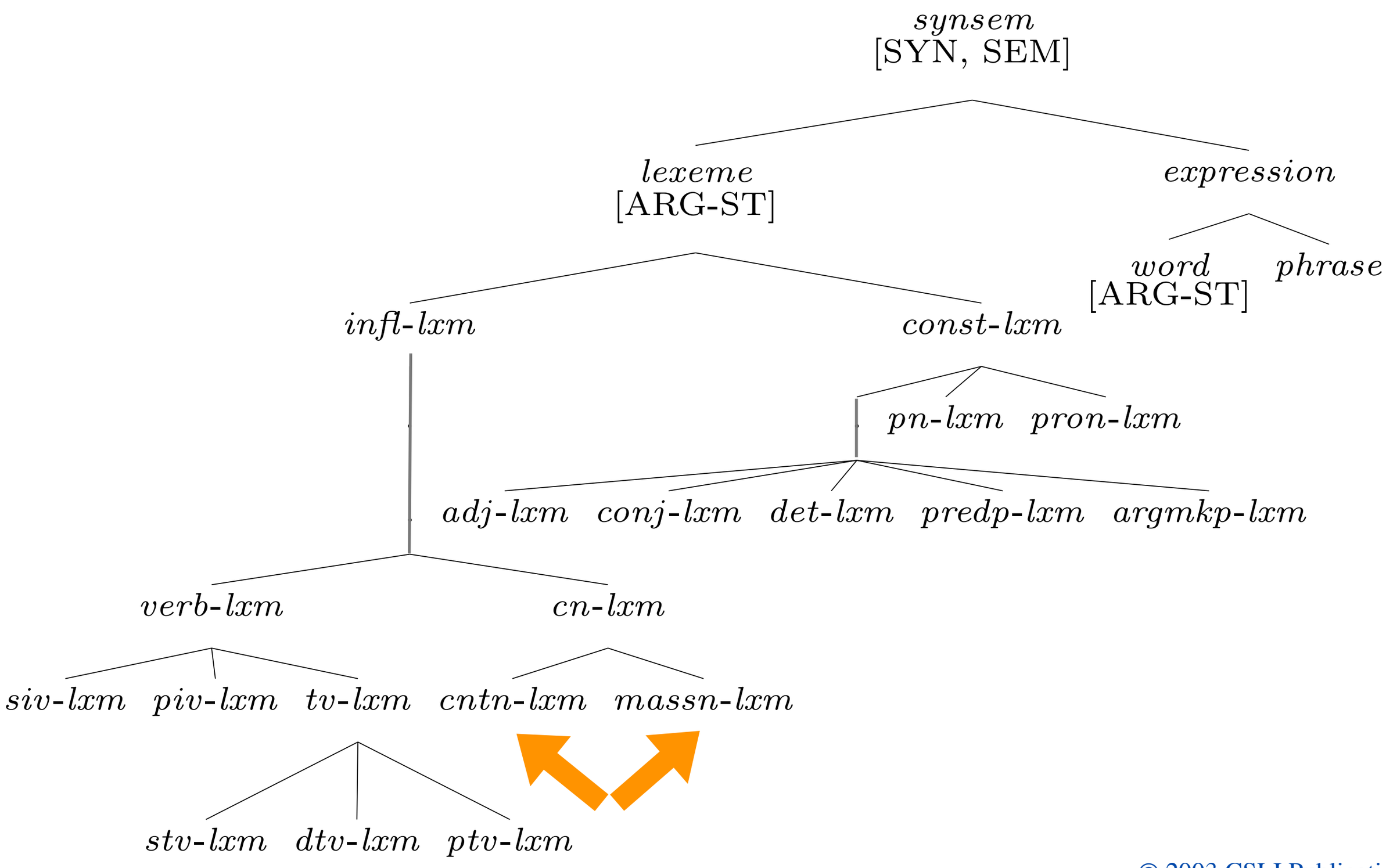
Constraints on *cn-lxm*



Constraints on *cn-lxm*

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

Formally Distinguishing Count vs. Mass Nouns

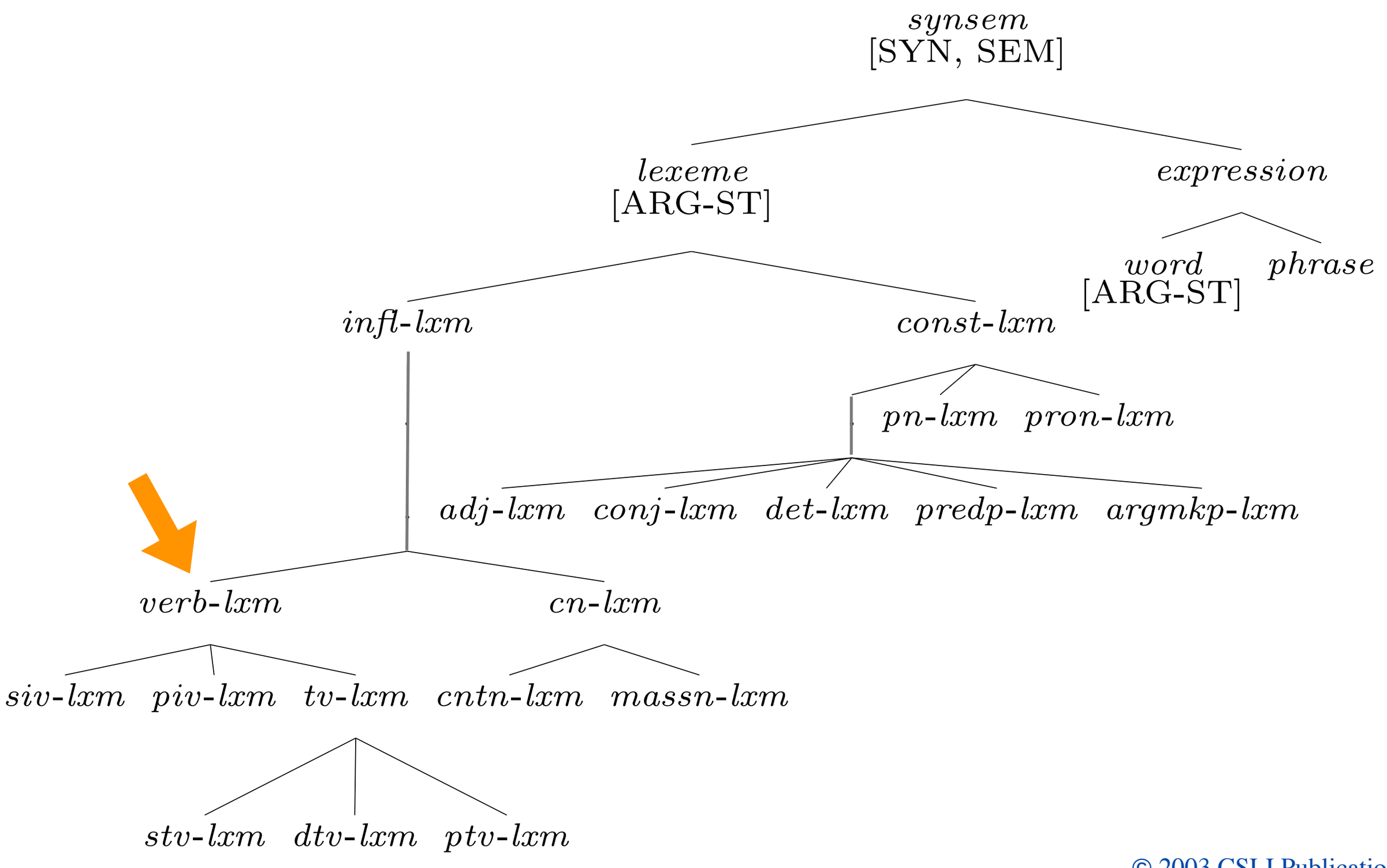


Formally Distinguishing Count vs. Mass Nouns

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

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

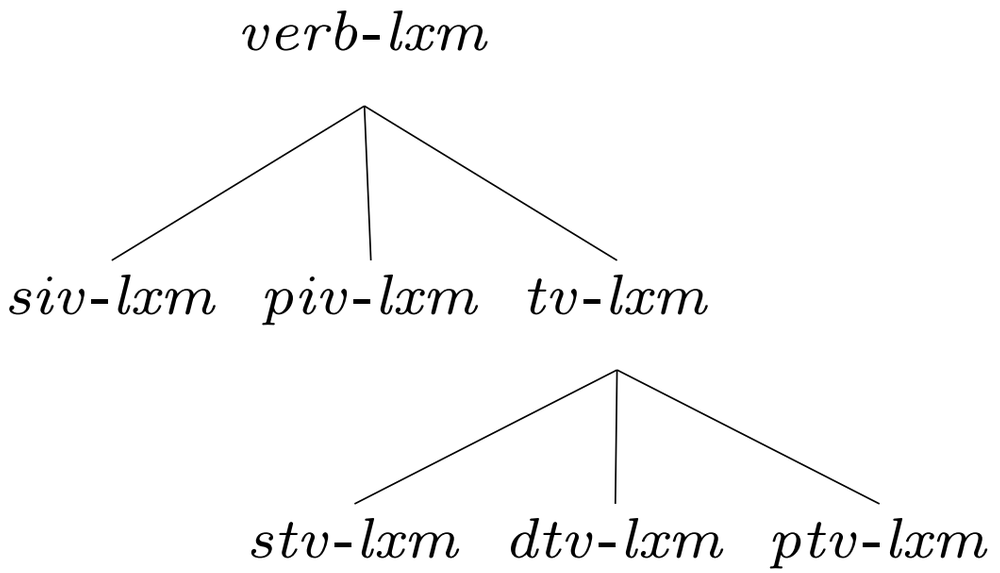
Constraints on *verb-lxm*



Constraints on *verb-lxm*

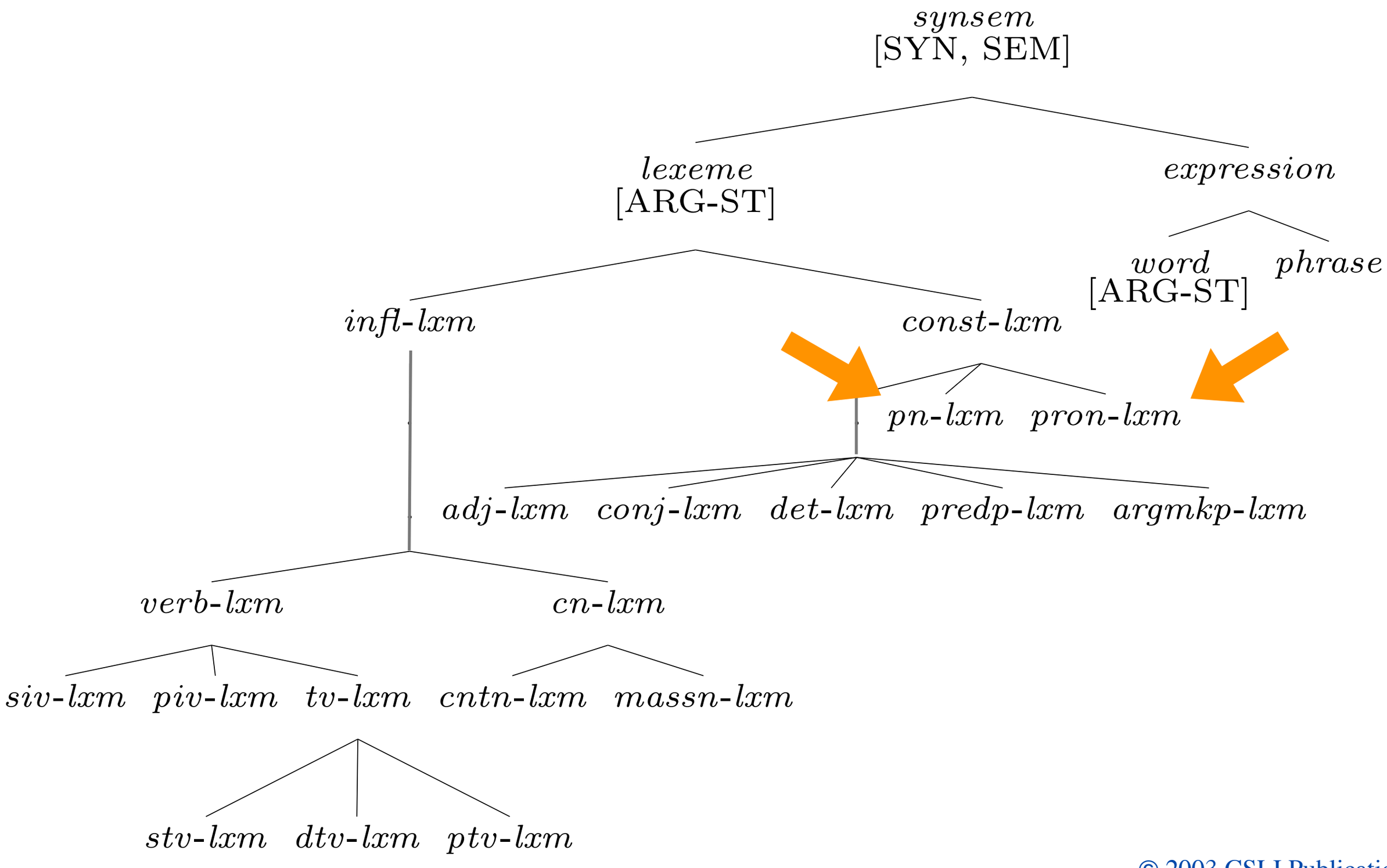
$$\textit{verb-lxm}\text{:}\quad \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \textit{verb} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{MODE} & \text{prop} \end{bmatrix} \\ \text{ARG-ST} & / \langle\ \text{NP},\ \ldots\ \rangle \end{bmatrix}$$

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*,...

Overview

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

Reading Questions

- What's the difference between a lexeme and a lemma?
- What's the relationship between lexical entry, lexical sequence, and word?
- What are X, Y and Z?

Reading Questions

- Could we use multiple inheritance to bring together all of the nouny lexical types and say [HEAD noun] just once?
- Why don't we want the SHAC to apply to proper nouns and pronouns as well as common nouns?

Reading Questions

- Why do the lexical types talk about ARG-ST instead of SPR/COMPS?
- How do we end up with an empty COMPS list for cn-lxm?
- Where do we encode the constraint that SPR is empty or has one thing on it?

Constraints on *cn-lxm*

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

Reading Questions

- Are only subtypes allowed to override defeasible constraints, or can individual lexemes do this as well? If so, how many words must have the same overriding rule for the words to be considered a new subtype? Is it a completely arbitrary decision?

- Was there a reason for choosing to mark constraints as defeasible vs. marking the constraints that are not defeasible? Are defeasible constraints more marked/rare in language and therefore deserve special notation? What is the motivation for even marking a constraint as defeasible (or not) at all?