Ling 566
Oct 29, 2015

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'

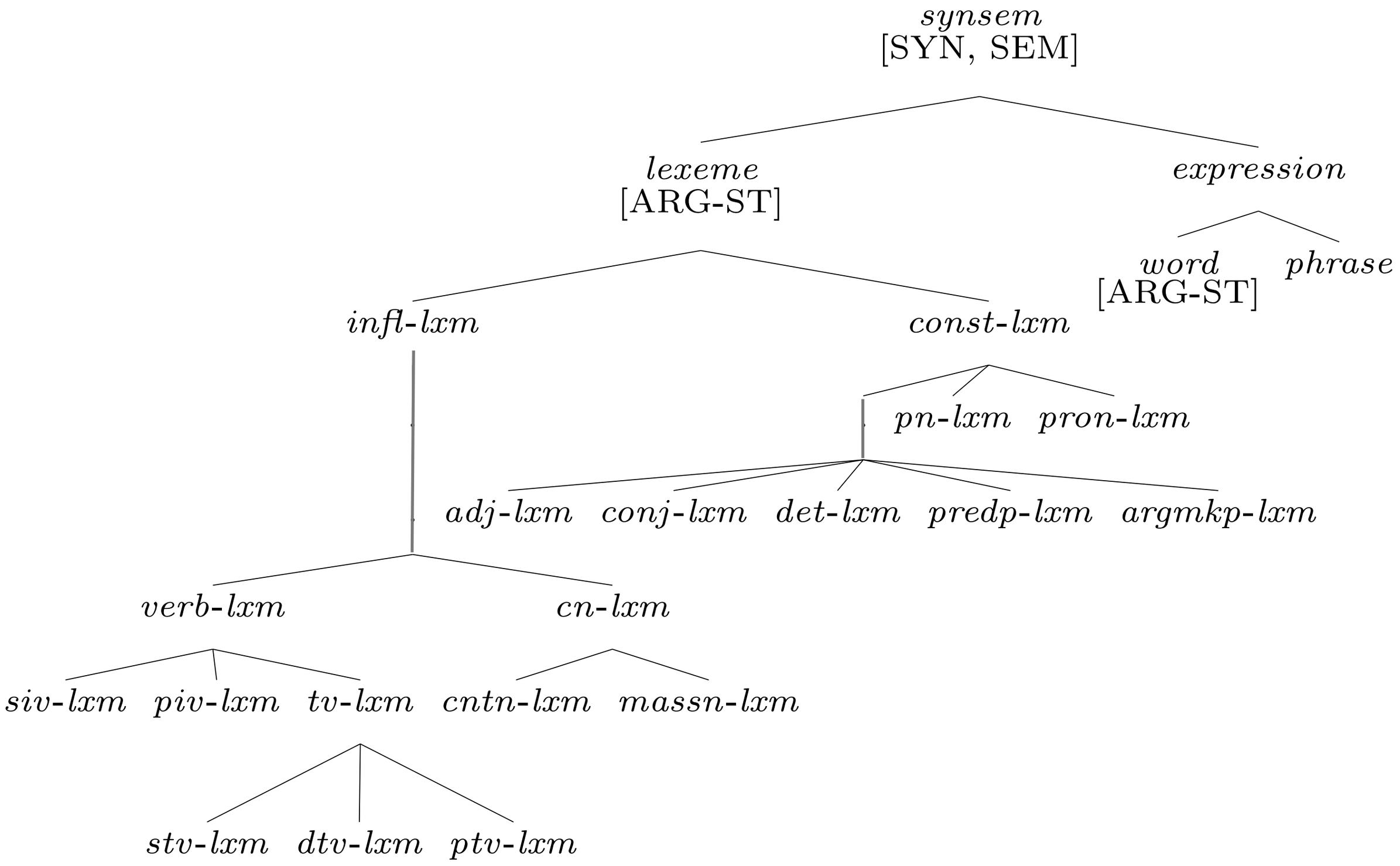
Question on Default Inheritance

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.

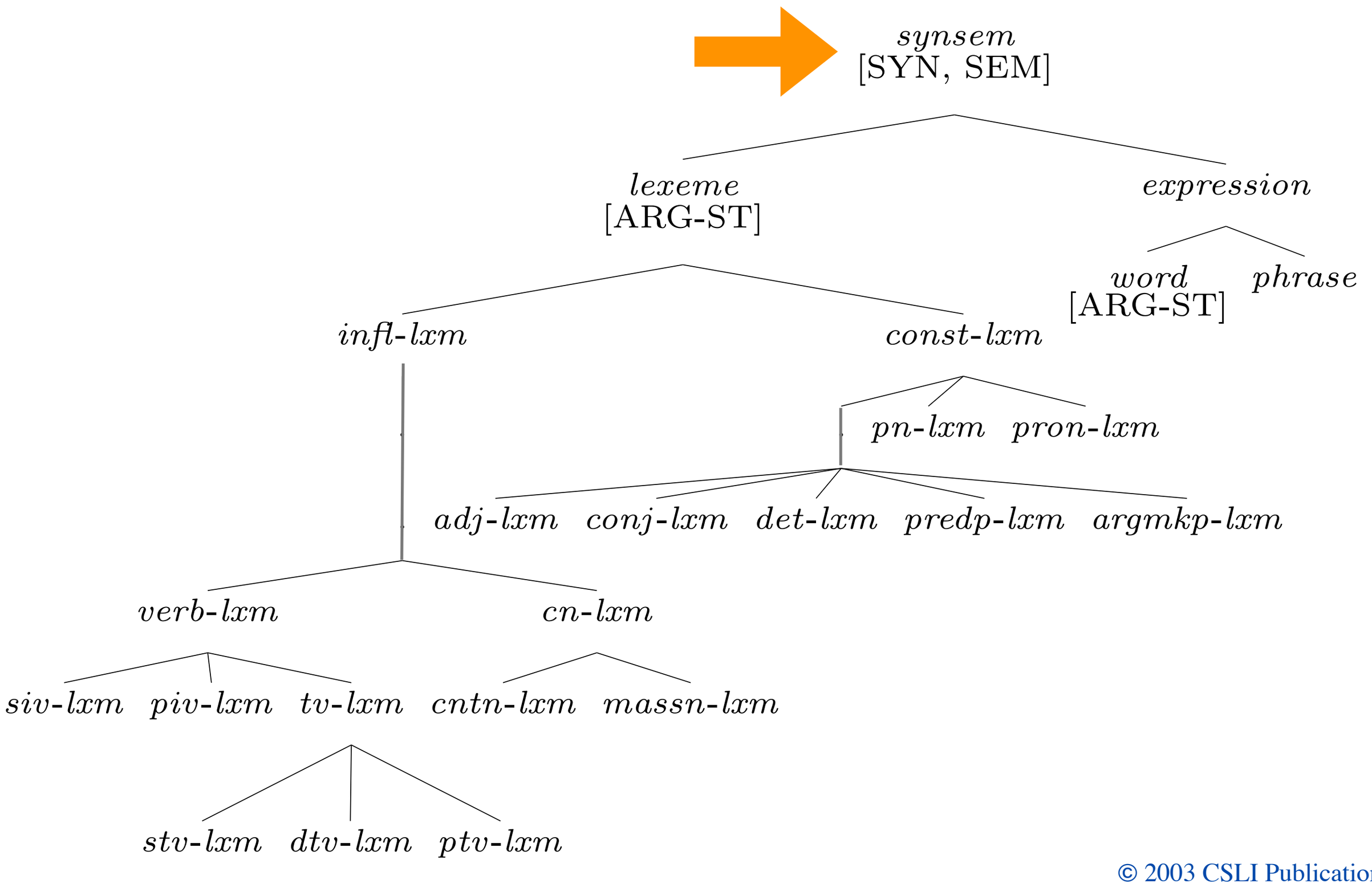
Our Lexeme Hierarchy



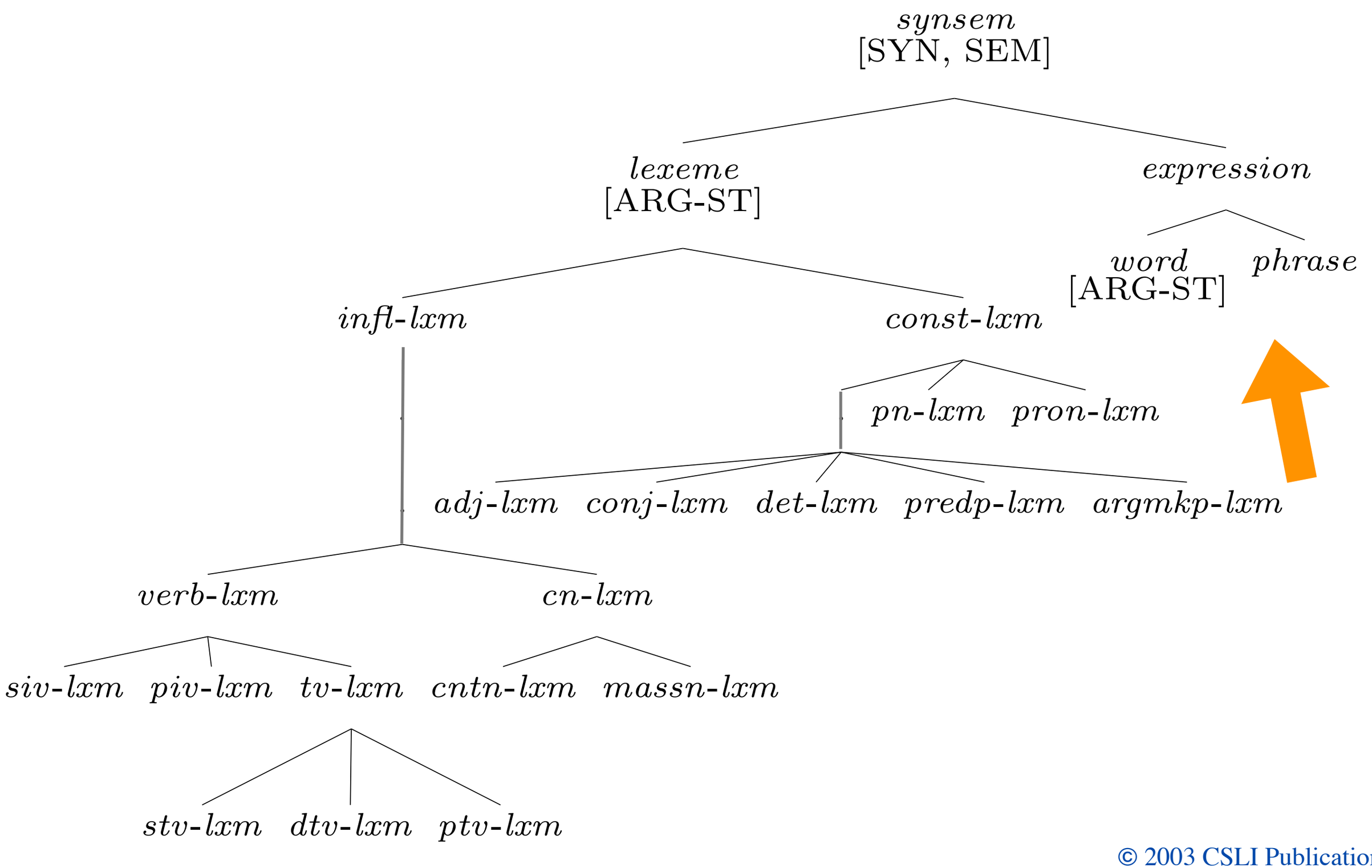
Functions of Types

- 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

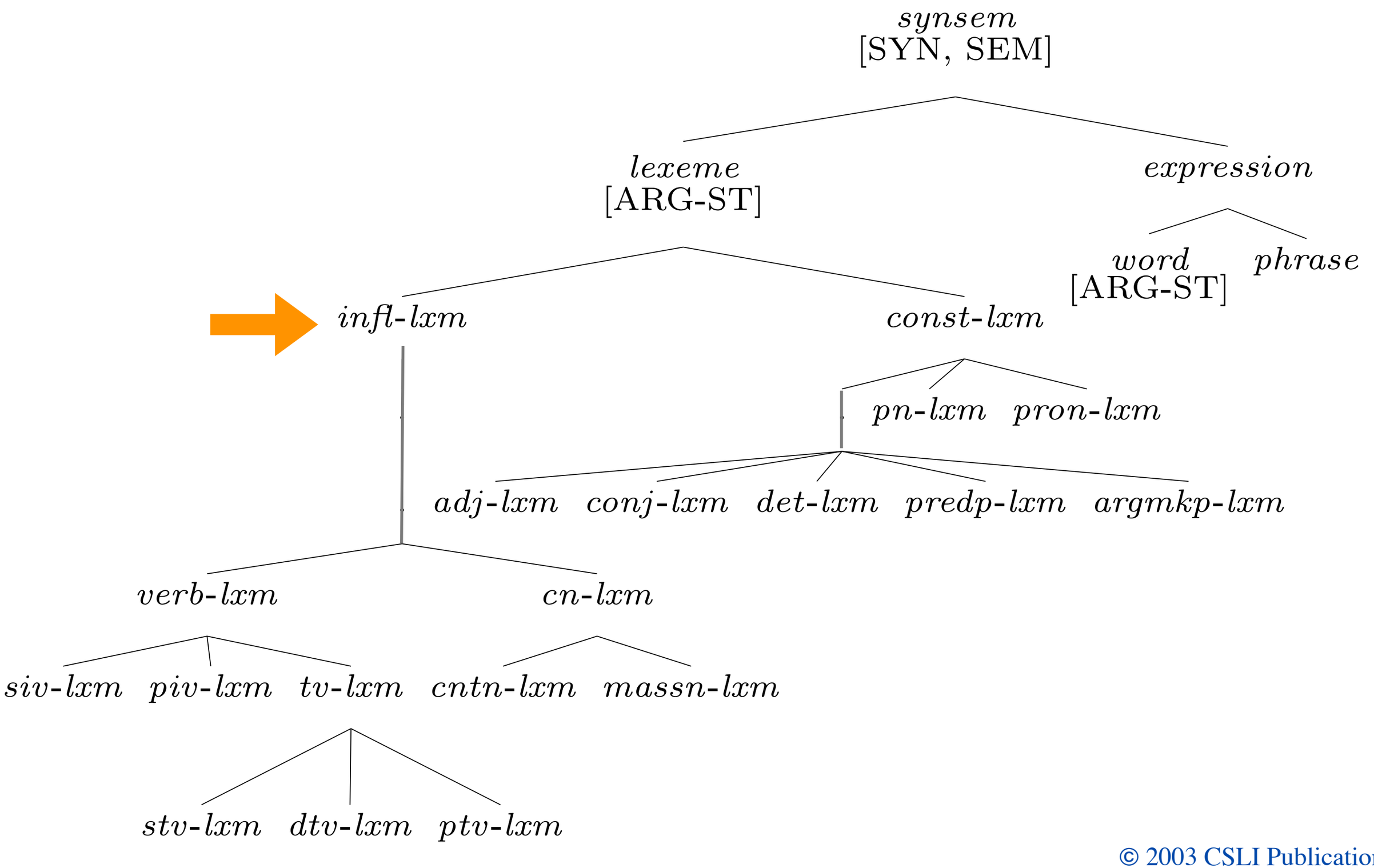
Every *synsem* has the features SYN and SEM



No ARG-ST on *phrase*



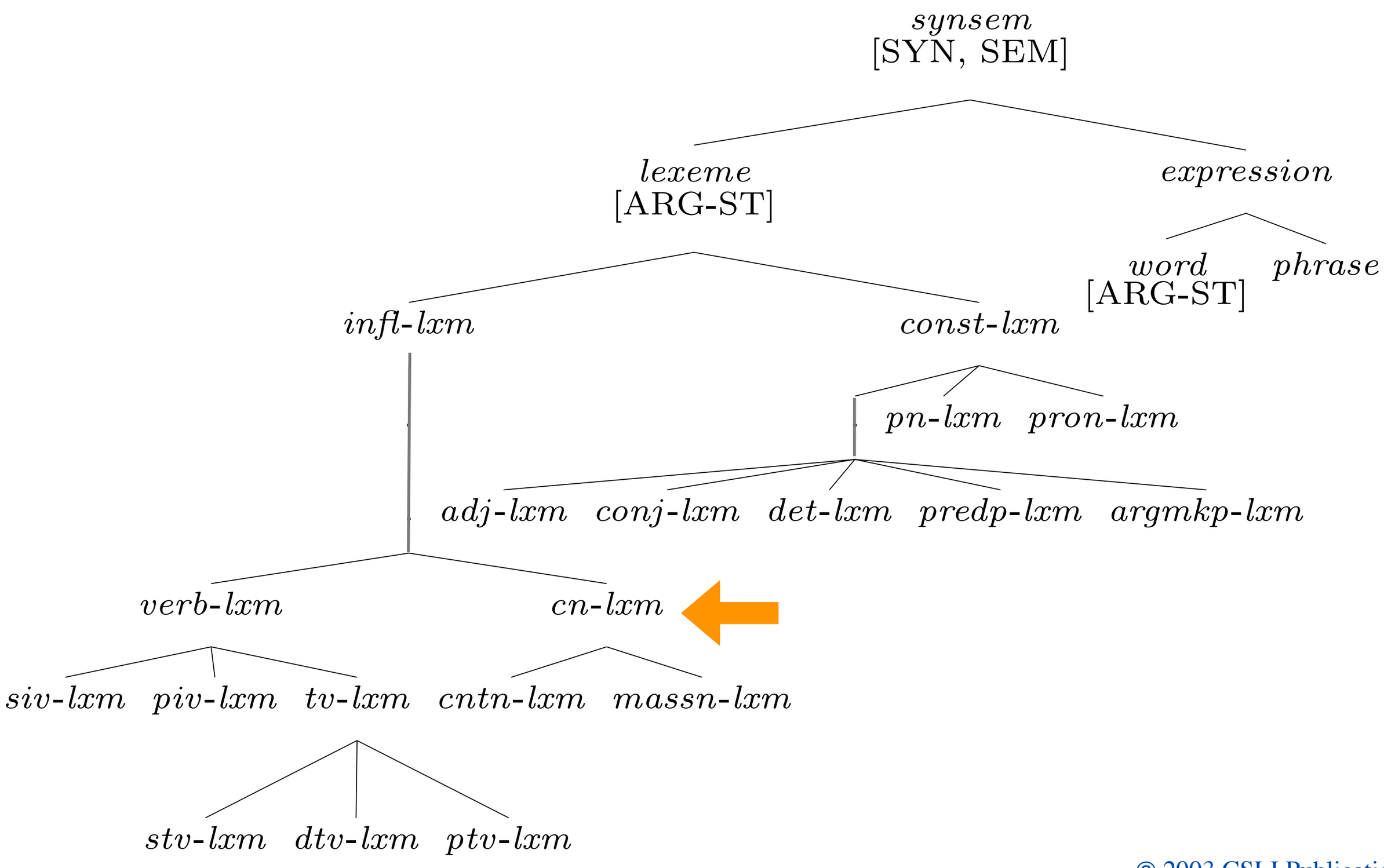
A Constraint on *infl-lxm*: the SHAC



A Constraint on *infl-lxm*: the SHAC

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

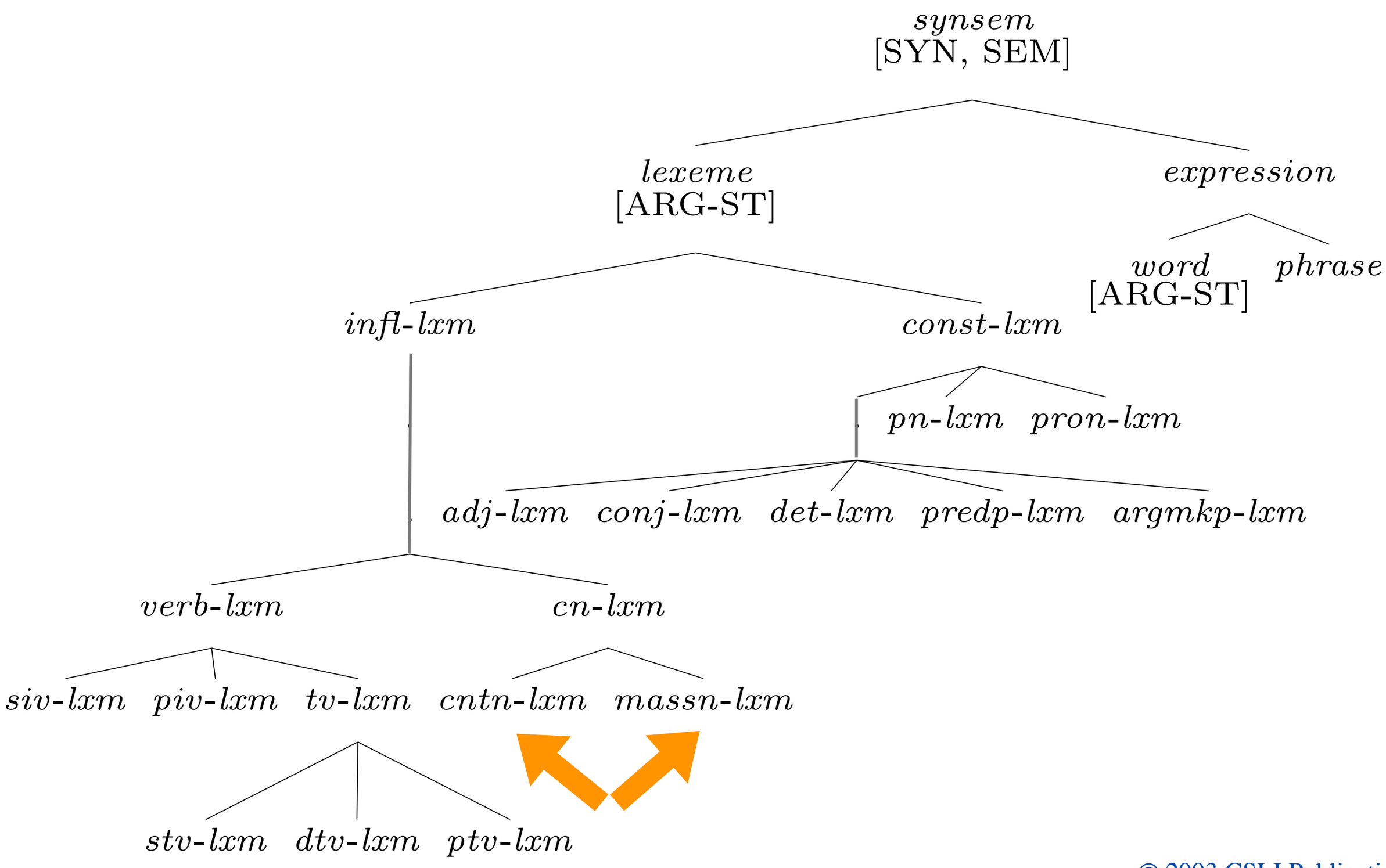
Constraints on *cn-lxm*



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

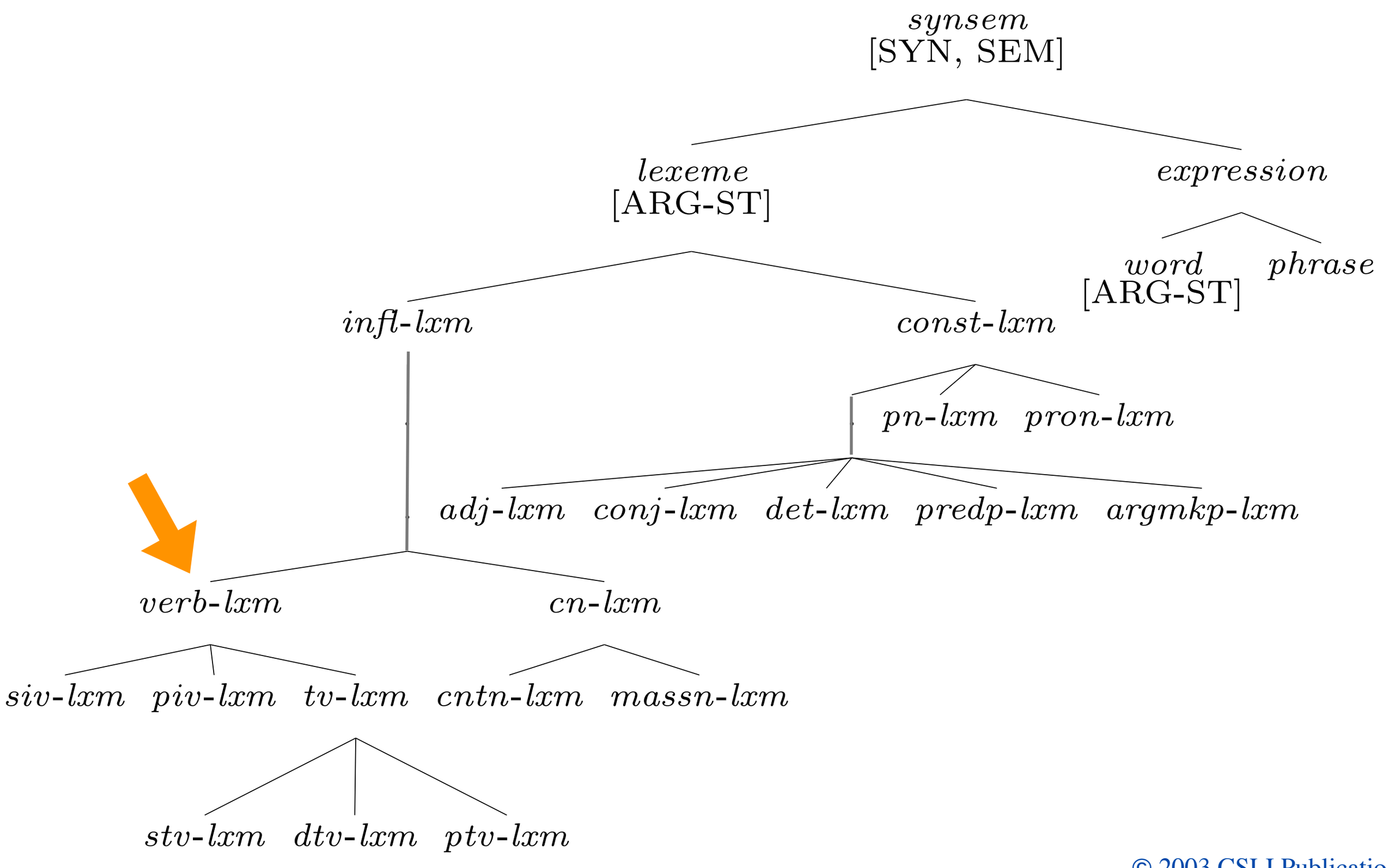


Formally Distinguishing Count vs. Mass Nouns

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

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

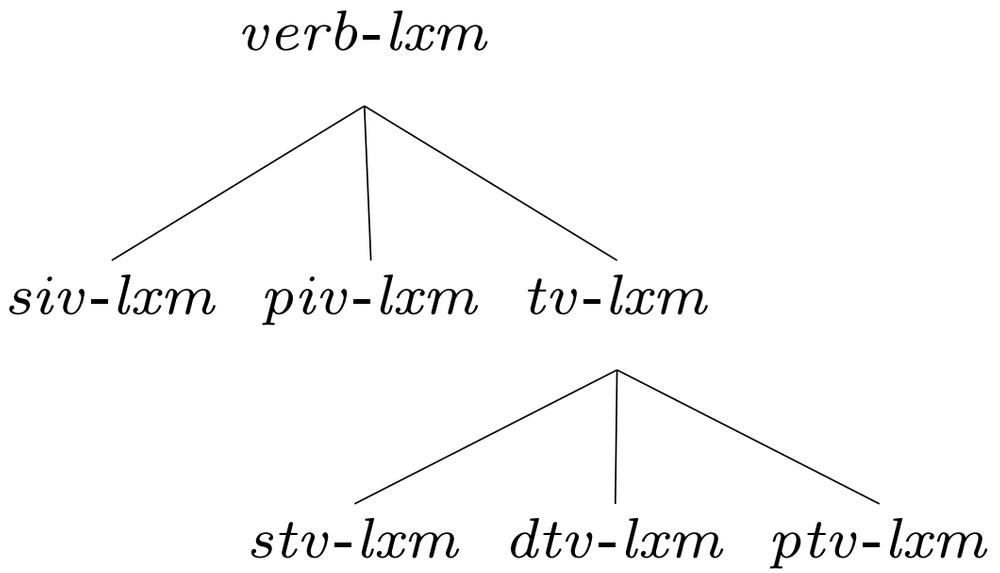
Constraints on *verb-lxm*



Constraints on *verb-lxm*

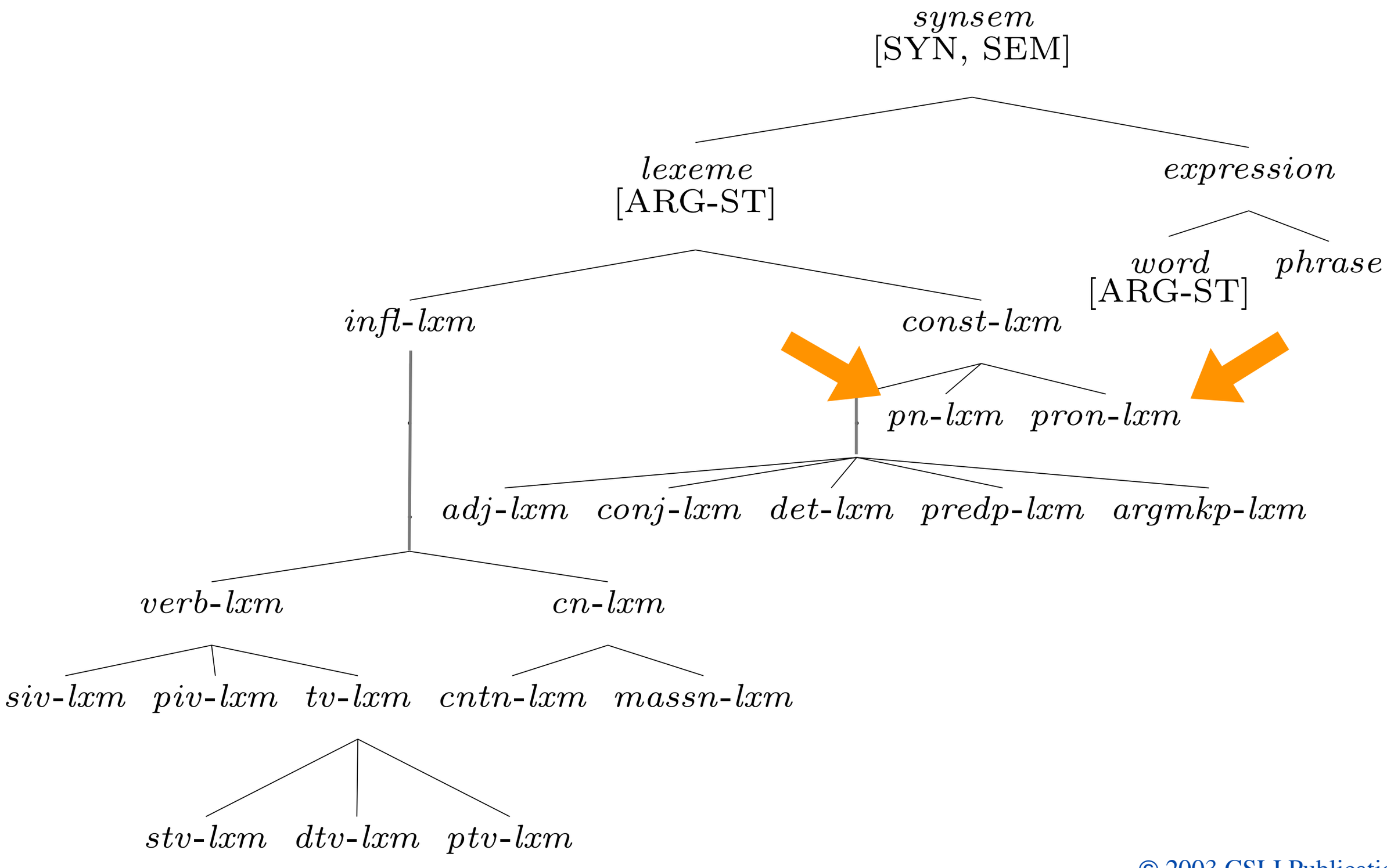
$$\textit{verb-lxm}\text{:}\quad \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \textit{verb} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{MODE} & \text{prop} \end{bmatrix} \\ \text{ARG-ST} & / \langle\ \text{NP},\ \ldots\ \rangle \end{bmatrix}$$

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 would the SPR of a predicative preposition or adjective be?
- Why is it that the "non-empty MOD value is irrelevant" (pg 243) when the preposition appears as a complement? Isn't it problematic that a complement preposition still has an unrealized MOD value?
- Would we consider the prepositions of phrasal verbs ("*She takes after her mother.*") to be of type argument-marking-preposition? Or is this something we haven't handled yet in HPSG?

Reading Questions

(41) a.

$$\textit{predp-lxm}: \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \textit{prep} \\ \text{VAL} & \begin{bmatrix} \text{SPR} & \langle\ \text{X}\ \rangle \\ \text{MOD} & \langle\ \text{Y}\ \rangle \end{bmatrix} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{MODE} & \text{prop} \\ \text{RESTR} & \langle\ \text{Z}\ \rangle \end{bmatrix} \\ \text{ARG-ST} & \langle\ \text{NP}\ ,\ \text{NP}\ \rangle \end{bmatrix}$$

Reading Questions

- What's the value of having defeasible constraints, if it's all cached out by the time you get to lexical entries?
- when authoring a grammar, is there a clear line between "default behavior" and simply "majority behavior" ? because if default behavior is always determined by the majority of lexemes, shouldn't the top basic lexeme entry have a default definition for a noun, because "the majority of words are nouns"?

Reading Questions

- I understand that in practice some constraints will NOT be overridden in any subtypes of a given lexeme, but why do we take pains to indicate when something CAN be overridden, instead of just taking as a general principle that all instances of a type inherit the constraints of that type unless their entry say otherwise?

- "Note that the default part of the constraint has been 'pushed down' to the next level of embedding in such a way as to have the maximum effect that is still consistent with the overriding constraint." How does one determine maximum effect and if it is still consistent with the overriding constraint?

Reading Questions

- It looks like *infl-lxm* is really just a type to accommodate the SHAC. Since the SHAC definition in previous chapters said that the constraint only applied to common nouns and verbs, I'm not entirely sure what moving it in the hierarchy is going to get us beyond what we already had. I guess we no longer have to say "for common nouns and verbs" in the definition, but is there something deeper I am missing?

Reading Questions

- Why exactly is the SHAC Constraint, as specified on page 238, not a defeasible constraint? Wouldn't it make sense to define the constraint such that the values of both AGR features are able to be overridden?

- The type verb-lxm requires all instances to have an NP at the start of its AGR-ST list. In the last class we talked about how certain verbs in imperative sentences are truly subjectless. How does this lexeme handle those verbs?

Reading Questions

- For (41a), we're told the MOD can be irrelevant. If so, why does it appear as non-defeasible in the type?
- Why does adj have [MODE prop] while adv has [MODE none]?

Reading Questions

- What does it mean when it says "ARG-ST < NP, ... >"?
- Does the verbal lexeme hierarchy allow us to add new branches for new structures, or would we need to fit these other types into the existing types with modifications where necessary? For example, if a verb takes a sentence complement, would this structure belong to *verb-lxm* or somewhere under *tv-lxm*?

Reading Questions

- Way back in Chapter 2 (right?), we originally identified verbs as simply being [COMPS itr], [COMPS str], etc. We then quickly realized that, given the large variety of valence patterns in English, that naming each pattern as a separate value of COMPS was an inelegant solution. Haven't we done the same thing by specifying valence patterns in our lexeme type hierarchy? Aren't we just going to end up exploding our type hierarchy into something just as unwieldy as what we had in Chapter 2?

Reading Questions

- It seems as though it would make sense to define an inheritance ("is-a") relationship between lexeme and word. Intuitively, runs and ran seem like children/subtypes of the supertype ran.

- Do the types of semantic relations between lexical items found in things like WordNet have a place in an HPSG theory of the lexicon? I was wondering in particular if the inheritance-based view from the chapter could be extended to account for the hypernym/hyponym relation (a dog is a canine is an animal, etc.).

Reading Questions

- What exactly is the difference between a lexical entry and a lexical sequence?
- What is an example of a lexical sequence which is not a lexical entry?
- What is the relationship between a lexical sequence and a lexeme? Is a lexical sequence a set of lexical entries that can be generated by a lexeme?

Reading Questions

lexical sequence Ordered pairs that can serve as the INPUT and OUTPUT values of lexical rules [q.v.] are called lexical sequences. They consist of a phonological form and a fully resolved feature structure.

lexical entry Information about individual words [q.v.] that must be stipulated is put into the lexicon [q.v.] in the form of descriptions that we call lexical entries. They are ordered pairs, consisting of a phonological form (description) and a partial feature structure description. Fully resolved lexical sequences [q.v.] consistent with lexical entries can serve as the INPUT values of lexical rules [q.v.].

lexeme The term 'word' is used ambiguously to mean either a particular form, such as *sees*, or a set of related forms such as *see, sees, saw, seen*, and *seeing*. To avoid this ambiguity, linguists sometimes posit an abstract entity called a 'lexeme' that gives rise to a family of related words. *See also* word.

Reading Questions

- Footnote 11 in section 8.4 states that a lexical entry is a "description", while a lexical sequence is a "model". What exactly does this mean? Conceptually I suppose a description is a static entry of information about an object, while a model can be manipulated to imitate behavior. But how does that apply to the notions of 'lexical entry" and "lexical sequence"?

Reading Questions

- In the past we've discussed both top-down and bottom-up in reference to information processing. I'm just curious about whether it's more common/beneficial to think of the lexicon bottom-up or not. It seems intuitive to focus on the lexeme first and work down to the word, but is it actually easier to focus on the word, and work your way up, in order to not waste time looking at default constraints that would just be contradicted closer to word?