Ling 566
Oct 5, 2017

Feature Structures
Headed Rules, Trees

Overview

- Review: problems with CFG, modeling
- Feature structures, unification (pizza)
- Features for linguistic description
- Reformulate grammar rules
- Notion of head/headedness
- Licensing of trees
- Reading questions

Our Goals

- Descriptive, generative grammar
  - Describing English (in this case)
  - Generating all possible well-formed sentences (and no ill-formed ones)
    - Assigning appropriate structures
- Design/discover an appropriate *type* of model (through incremental improvement)
- Create a particular model (grammar fragment) for English

Problems with Context-Free Grammar (atomic node labels)

- Potentially arbitrary rules
- Gets clunky quickly with cross-cutting properties
- Not quite powerful enough for natural languages

Solution: Replace atomic node labels with feature structures.

Cross-cutting Grammatical Properties

	3rd singular subject	plural subject
direct object NP	*denies*	*deny*
no direct object NP	*disappears*	*disappear*

Two Kinds of Language Models

- Speakers' internalized knowledge (their grammar)
- Set of sentences in the language

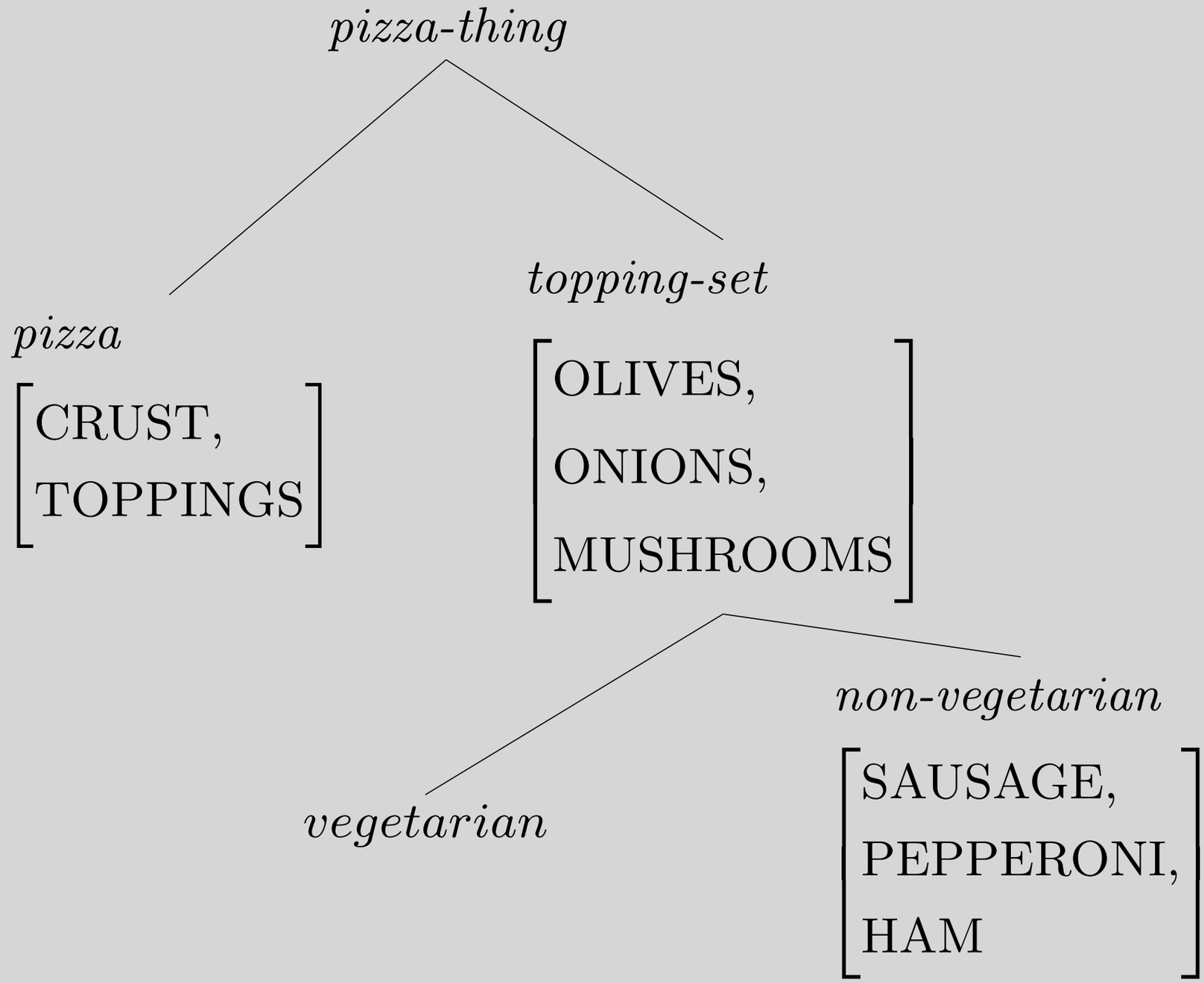
Things Involved in Modeling Language

- Real world entities (utterance types)
- Models (fully specified trees)
- Descriptions of the models (rules, principles, lexical entries)

Feature Structure Descriptions

$$\begin{bmatrix} \text{FEATURE}_1 & \text{VALUE}_1 \\ \text{FEATURE}_2 & \text{VALUE}_2 \\ \ldots & \\ \text{FEATURE}_n & \text{VALUE}_n \end{bmatrix}$$

A Pizza Type Hierarchy



TYPE	FEATURES/VALUES	IST
pizza-thing		
pizza	$\begin{bmatrix} \text{CRUST} & \{\text{thick, thin, stuffed}\} \\ \text{TOPPINGS} & \textit{topping-set} \end{bmatrix}$	*pizza-thing*
topping-set	$\begin{bmatrix} \text{OLIVES} & \{+, -\} \\ \text{ONIONS} & \{+, -\} \\ \text{MUSHROOMS} & \{+, -\} \end{bmatrix}$	*pizza-thing*
vegetarian		*topping-set*
non-vegetarian	$\begin{bmatrix} \text{SAUSAGE} & \{+, -\} \\ \text{PEPPERONI} & \{+, -\} \\ \text{HAM} & \{+, -\} \end{bmatrix}$	*topping-set*

Type Hierarchies

A type hierarchy....

- ... states what kinds of objects we claim exist (the types)
- ... organizes the objects hierarchically into classes with shared properties (the type hierarchy)
- ... states what general properties each kind of object has (the feature and feature value declarations).

Pizza Descriptions and Pizza Models

$$\begin{bmatrix} \textit{pizza} & \\ \text{CRUST} & \text{thick} \\ \text{TOPPINGS} & \begin{bmatrix} \textit{vegetarian} & \\ \text{OLIVES} & + \\ \text{ONIONS} & + \end{bmatrix} \end{bmatrix}$$

How many pizza models (by definition, fully resolved) satisfy this description?

Answer: 2

$$\begin{bmatrix} \textit{pizza} & \\ \text{CRUST} & \text{thick} \\ \text{TOPPINGS} & \begin{bmatrix} \textit{vegetarian} & \\ \text{OLIVES} & + \\ \text{ONIONS} & + \end{bmatrix} \end{bmatrix}$$

{<CRUST , thick> , <TOPPINGS , { <OLIVES , + > , <ONIONS, +> , <MUSHROOMS, –>}>}

{<CRUST , thick> , <TOPPINGS , { <OLIVES , + > , <ONIONS, +> , <MUSHROOMS, +>}>}

Pizza Descriptions and Pizza Models

$$\begin{bmatrix} \textit{pizza} & \\ \text{CRUST} & \text{thick} \\ \text{TOPPINGS} & \begin{bmatrix} \textit{vegetarian} & \\ \text{OLIVES} & + \\ \text{ONIONS} & + \end{bmatrix} \end{bmatrix}$$

How many pizzas-in-the-world do the pizza models correspond to?

Answer: A large, constantly-changing number.

Pizza Descriptions and Pizza Models

$$\begin{bmatrix} \textit{pizza} & \\ \text{CRUST} & \text{thick} \\ \text{TOPPINGS} & \begin{bmatrix} \textit{vegetarian} & \\ \text{OLIVES} & + \\ \text{ONIONS} & + \end{bmatrix} \end{bmatrix}$$

‘type’/‘token’ distinction
applies to sentences as well

Combining Constraints

$$\begin{bmatrix} pizza \\ \text{CRUST} & \text{thick} \\ \text{TOPPINGS} & \begin{bmatrix} \text{OLIVES} & + \\ \text{HAM} & - \end{bmatrix} \end{bmatrix} \;\&\; \begin{bmatrix} pizza \\ \text{TOPPINGS} & \begin{bmatrix} \text{OLIVES} & + \\ \text{ONIONS} & + \end{bmatrix} \end{bmatrix}$$

Combining Constraints

$$\begin{bmatrix} \textit{pizza} & \\ \text{CRUST} & \text{thick} \\ \text{TOPPINGS} & \begin{bmatrix} \text{OLIVES} & + \\ \text{ONIONS} & + \\ \text{HAM} & - \end{bmatrix} \end{bmatrix}$$

Combining Constraints

$$\begin{bmatrix} pizza \\ \text{CRUST} & \text{thick} \\ \text{TOPPINGS} & \begin{bmatrix} \text{OLIVES} & + \\ \text{HAM} & - \end{bmatrix} \end{bmatrix} \;\&\; \begin{bmatrix} pizza \\ \text{CRUST} & \text{thin} \\ \text{TOPPINGS} & \begin{bmatrix} \text{OLIVES} & + \\ \text{ONIONS} & + \end{bmatrix} \end{bmatrix}$$

$$= \phi$$

Combining Constraints

$$\begin{bmatrix} pizza \\ \text{CRUST} & \text{thick} \\ \text{TOPPINGS} & \begin{bmatrix} \text{OLIVES} & + \\ \text{HAM} & + \end{bmatrix} \end{bmatrix} \;\&\; \begin{bmatrix} pizza \\ \text{CRUST} & \text{thick} \\ \text{TOPPINGS} & vegetarian \end{bmatrix}$$

$$= \phi$$

Combining Constraints

$$\begin{bmatrix} \textit{pizza} \\ \text{CRUST} & \text{thick} \\ \text{TOPPINGS} & \begin{bmatrix} \text{OLIVES} & + \\ \text{HAM} & - \end{bmatrix} \end{bmatrix} \;\&\; \begin{bmatrix} \textit{pizza} \\ \text{CRUST} & \text{thick} \\ \text{TOPPINGS} & \textit{vegetarian} \end{bmatrix}$$

$$= \phi$$

A New Theory of Pizzas

$$pizza: \begin{bmatrix} \text{CRUST} & \{\text{thick , thin , stuffed}\} \\ \text{ONE-HALF} & \textit{topping-set} \\ \text{OTHER-HALF} & \textit{topping-set} \end{bmatrix}$$

Combining Constraints

$$\begin{bmatrix} pizza \\ \text{ONE-HALF} & \begin{bmatrix} \text{ONIONS} & + \\ \text{OLIVES} & - \end{bmatrix} \end{bmatrix} \;\&\; \begin{bmatrix} pizza \\ \text{OTHER-HALF} & \begin{bmatrix} \text{ONIONS} & - \\ \text{OLIVES} & + \end{bmatrix} \end{bmatrix}$$

$$=$$

$$\begin{bmatrix} pizza & \\ \text{ONE-HALF} & \begin{bmatrix} \text{ONIONS} & + \\ \text{OLIVES} & - \end{bmatrix} \\ \text{OTHER-HALF} & \begin{bmatrix} \text{ONIONS} & - \\ \text{OLIVES} & + \end{bmatrix} \end{bmatrix}$$

Identity Constraints (tags)

$$\begin{bmatrix} \textit{pizza} & \\ \text{CRUST} & \text{thin} \\ \text{ONE-HALF} & \begin{bmatrix} \text{OLIVES} & \boxed{1} \\ \text{ONIONS} & \boxed{2} \end{bmatrix} \\ \text{OTHER-HALF} & \begin{bmatrix} \text{OLIVES} & \boxed{1} \\ \text{ONIONS} & \boxed{2} \end{bmatrix} \end{bmatrix}$$

Combining Constraints

$$\begin{bmatrix} pizza \\ \text{ONE-HALF} & \boxed{1}\begin{bmatrix}\text{ONIONS} & + \\ \text{OLIVES} & -\end{bmatrix} \\ \text{OTHER-HALF} & \boxed{1} \end{bmatrix} \;\&\; \begin{bmatrix} pizza \\ \text{OTHER-HALF} & \begin{bmatrix}\text{MUSHROOMS} & - \\ \text{OLIVES} & -\end{bmatrix} \end{bmatrix}$$

$$=$$

$$\begin{bmatrix} pizza \\ \text{ONE-HALF} & \boxed{1}\begin{bmatrix}\text{ONIONS} & + \\ \text{OLIVES} & - \\ \text{MUSHROOMS} & -\end{bmatrix} \\ \text{OTHER-HALF} & \boxed{1} \end{bmatrix}$$

Note

$$\begin{bmatrix} \textit{pizza} & \\ \text{ONE-HALF} & \boxed{1}\begin{bmatrix} \text{ONIONS} & + \\ \text{OLIVES} & - \\ \text{MUSHROOMS} & - \end{bmatrix} \\ \text{OTHER-HALF} & \boxed{1} \end{bmatrix}$$

$$=$$

$$\begin{bmatrix} \textit{pizza} & \\ \text{ONE-HALF} & \boxed{1} \\ \text{OTHER-HALF} & \boxed{1}\begin{bmatrix} \text{ONIONS} & + \\ \text{OLIVES} & - \\ \text{MUSHROOMS} & - \end{bmatrix} \end{bmatrix}$$

Combining Constraints

$$\begin{bmatrix} pizza \\ \text{ONE-HALF} & \boxed{1}\begin{bmatrix}\text{ONIONS} & + \\ \text{OLIVES} & +\end{bmatrix} \\ \text{OTHER-HALF} & \boxed{1}\ vegetarian \end{bmatrix} \;\&\; \begin{bmatrix} pizza \\ \text{ONE-HALF} & \begin{bmatrix}\text{SAUSAGE} & + \\ \text{HAM} & -\end{bmatrix} \end{bmatrix}$$

$$= \phi$$

Why combine constraints?

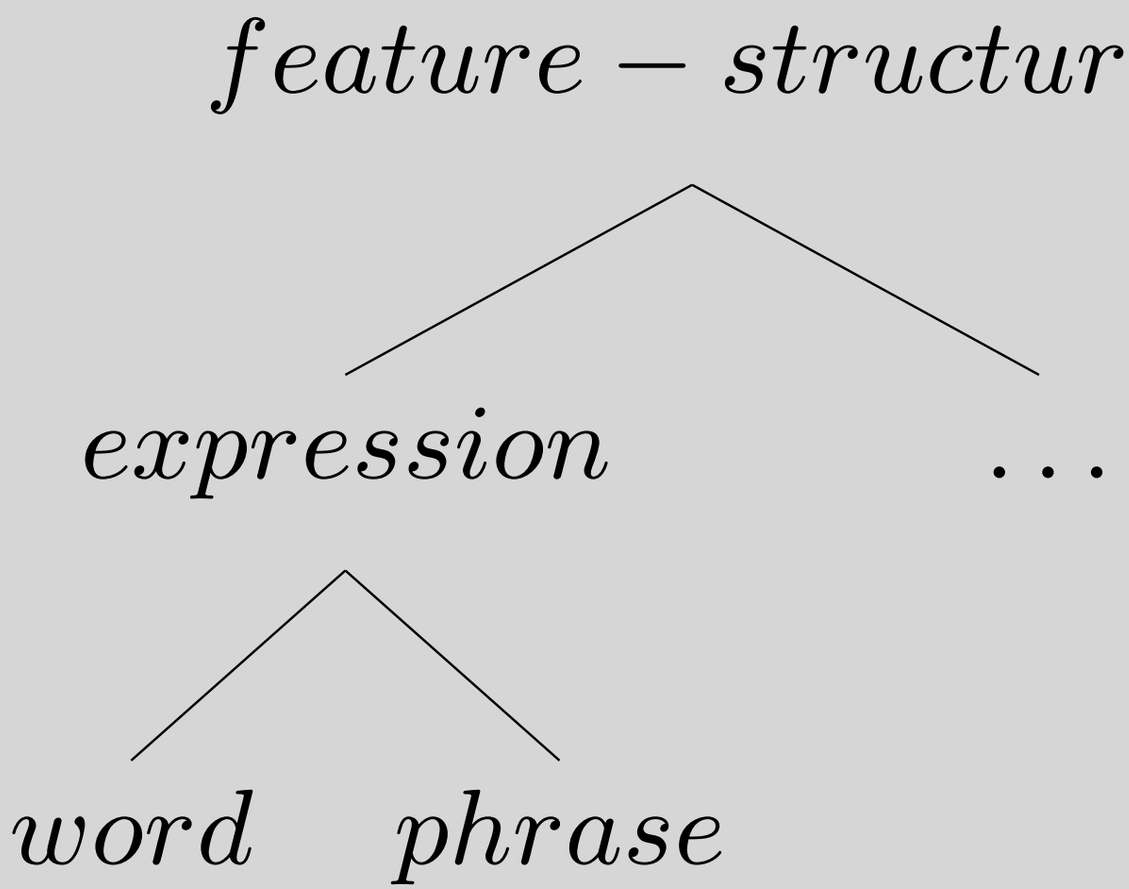
- The pizza example illustrates how unification can be used to combine information from different sources.
- In our grammar, information will come from lexical entries, grammar rules, and general principles.

Linguistic Application of Feature Structures: Making the Mnemonic Meaningful

What do these CFG categories have in common?

NP & VP:	are both phrases
N & V:	are both words
NP & N:	are both ‘nouny’
VP & V:	are both ‘verby’

The Beginnings of Our Type Hierarchy

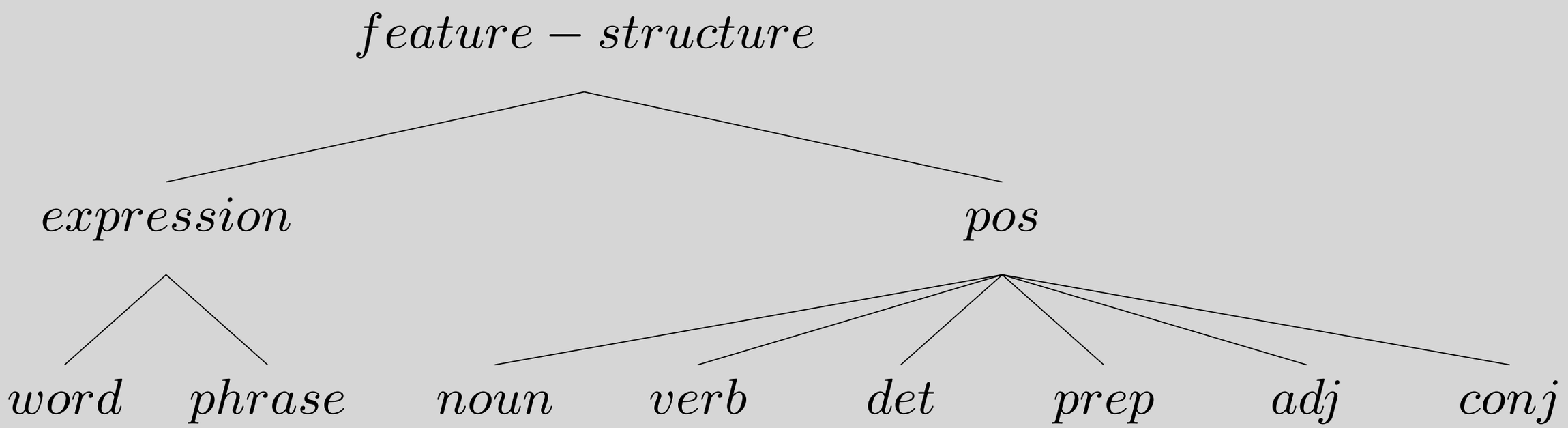


A Feature for Part of Speech

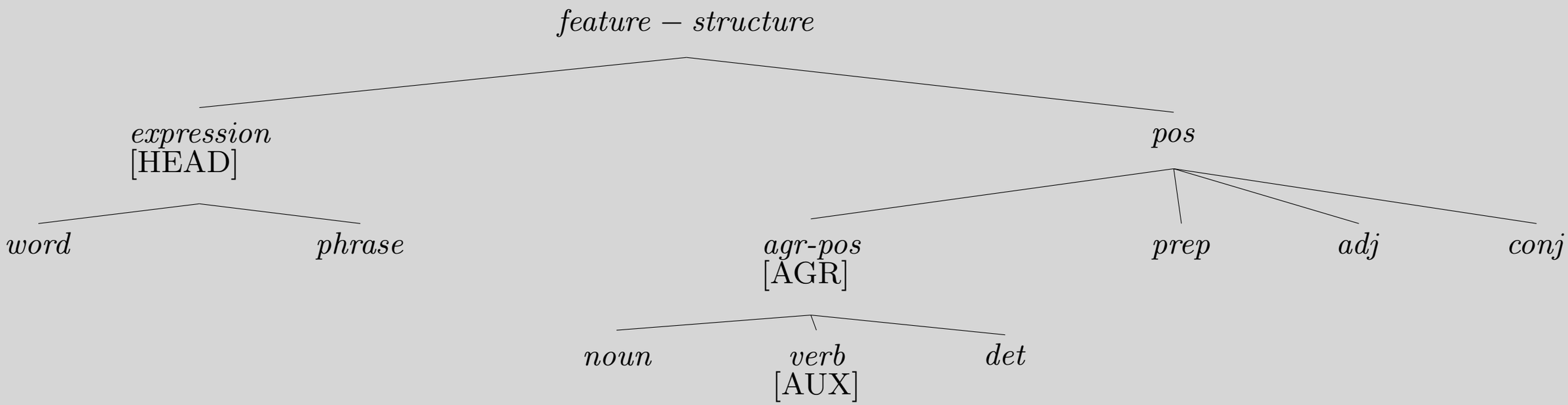
$$\text{NP} = \begin{bmatrix} \textit{phrase} & \\ \text{HEAD} & \textit{noun} \end{bmatrix}$$

$$\left\langle \text{bird} \ , \ \begin{bmatrix} \textit{word} & \\ \text{HEAD} & \textit{noun} \end{bmatrix} \right\rangle$$

Type Hierarchy for Parts of Speech I



Type Hierarchy for Parts of Speech II



A Feature for Valence

$$\text{IV} = \begin{bmatrix} \textit{word} & \\ \text{HEAD} & \textit{verb} \\ \text{VAL} & [\text{COMPS} \quad \text{itr}] \end{bmatrix}$$

$$\text{TV} = \begin{bmatrix} \textit{word} & \\ \text{HEAD} & \textit{verb} \\ \text{VAL} & [\text{COMPS} \quad \text{str}] \end{bmatrix}$$

$$\text{DTV} = \begin{bmatrix} \textit{word} & \\ \text{HEAD} & \textit{verb} \\ \text{VAL} & [\text{COMPS} \quad \text{dtr}] \end{bmatrix}$$

Underspecification

$$\mathrm{V} = \begin{bmatrix} word & \\ \mathrm{HEAD} & verb \end{bmatrix}$$

$$\mathrm{VP} = \begin{bmatrix} phrase & \\ \mathrm{HEAD} & verb \end{bmatrix}$$

$$[\mathrm{HEAD} \quad verb\ ]$$

Another Valence Feature

$$\text{NP} = \begin{bmatrix} \textit{phrase} & \\ \text{HEAD} & \textit{noun} \\ \text{VAL} & \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & + \end{bmatrix} \end{bmatrix}$$

$$\text{NOM} = \begin{bmatrix} \textit{phrase} & \\ \text{HEAD} & \textit{noun} \\ \text{VAL} & \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & - \end{bmatrix} \end{bmatrix}$$

SPR and Verbs

$$\mathrm{S} = \begin{bmatrix} \textit{phrase} & \\ \mathrm{HEAD} & \textit{verb} \\ \mathrm{VAL} & \begin{bmatrix} \mathrm{COMPS} & \mathrm{itr} \\ \mathrm{SPR} & + \end{bmatrix} \end{bmatrix}$$

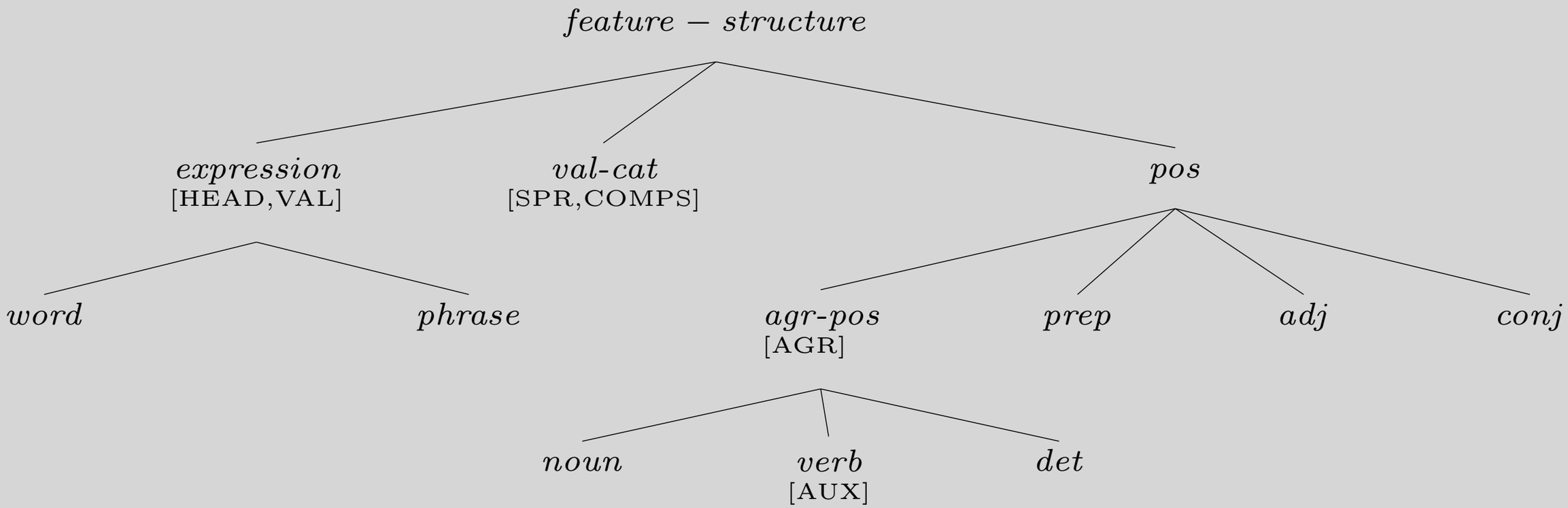
$$\mathrm{VP} = \begin{bmatrix} \textit{phrase} & \\ \mathrm{HEAD} & \textit{verb} \\ \mathrm{VAL} & \begin{bmatrix} \mathrm{COMPS} & \mathrm{itr} \\ \mathrm{SPR} & - \end{bmatrix} \end{bmatrix}$$

S and NP

$$\begin{bmatrix} \text{VAL} & \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & + \end{bmatrix} \end{bmatrix}$$

- We created a monster
- our creation of a monster

Type Hierarchy So Far



Reformulating the Grammar Rules I
Which Ch 2 rules do these correspond to?

Head-Complement Rule 1:

$$\begin{bmatrix} \textit{phrase} \\ \text{VAL} \quad \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & - \end{bmatrix} \end{bmatrix} \rightarrow \mathbf{H} \begin{bmatrix} \textit{word} \\ \text{VAL} \quad \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & - \end{bmatrix} \end{bmatrix}$$

Head Complement Rule 2:

$$\begin{bmatrix} \textit{phrase} \\ \text{VAL} \quad \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & - \end{bmatrix} \end{bmatrix} \rightarrow \mathbf{H} \begin{bmatrix} \textit{word} \\ \text{VAL} \quad \begin{bmatrix} \text{COMPS} & \text{str} \\ \text{SPR} & - \end{bmatrix} \end{bmatrix} \text{ NP}$$

Head Complement Rule 3:

$$\begin{bmatrix} \textit{phrase} \\ \text{VAL} \quad \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & - \end{bmatrix} \end{bmatrix} \rightarrow \mathbf{H} \begin{bmatrix} \textit{word} \\ \text{VAL} \quad \begin{bmatrix} \text{COMPS} & \text{dtr} \\ \text{SPR} & - \end{bmatrix} \end{bmatrix} \text{ NP NP}$$

Reformulating the Grammar Rules II

Head-Specifier Rule 1:

$$\begin{bmatrix} phrase \\ \text{VAL} \quad \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & + \end{bmatrix} \end{bmatrix} \rightarrow \overset{\text{NP}}{\begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{AGR} & \boxed{1} \end{bmatrix} \end{bmatrix}} \quad \mathbf{H} \begin{bmatrix} phrase \\ \text{HEAD} \quad \begin{bmatrix} verb \\ \text{AGR} \quad \boxed{1} \end{bmatrix} \\ \text{VAL} \quad \begin{bmatrix} \text{SPR} & - \end{bmatrix} \end{bmatrix}$$

Head-Specifier Rule 2:

$$\begin{bmatrix} phrase \\ \text{VAL} \quad \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & + \end{bmatrix} \end{bmatrix} \rightarrow \text{D} \quad \mathbf{H} \begin{bmatrix} phrase \\ \text{HEAD} & noun \\ \text{VAL} & \begin{bmatrix} \text{SPR} & - \end{bmatrix} \end{bmatrix}$$

Reformulating the Grammar Rules III

Non-Branching NP Rule

$$\begin{bmatrix} \textit{phrase} \\ \text{VAL} \quad \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & + \end{bmatrix} \end{bmatrix} \rightarrow \mathbf{H} \begin{bmatrix} \textit{word} \\ \text{HEAD} \quad \textit{noun} \\ \text{VAL} \quad \begin{bmatrix} \text{SPR} & + \end{bmatrix} \end{bmatrix}$$

Head-Modifier Rule

$$\begin{bmatrix} \textit{phrase} \\ \text{VAL} \quad \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & - \end{bmatrix} \end{bmatrix} \rightarrow \mathbf{H} \begin{bmatrix} \textit{phrase} \\ \text{VAL} \quad \begin{bmatrix} \text{SPR} & - \end{bmatrix} \end{bmatrix} \text{PP}$$

Coordination Rule

$$\boxed{1} \rightarrow \boxed{1}^{+} \begin{bmatrix} \textit{word} \\ \text{HEAD} \quad \textit{conj} \end{bmatrix} \boxed{1}$$

Advantages of the New Formulation

- Subject-verb agreement is stipulated only once (where?)
- Common properties of verbs with different valences are expressed by common features (for example?)
- Parallelisms across phrase types are captured (for example?)

Disadvantages of the New Formulation

- We still have three head complement rules
- We still have two head specifier rules
- We only deal with three verb valences (Which ones? What are some others?)
- The non-branching rule doesn't really do any empirical work
- Others?

Heads

- Intuitive idea: A phrase typically contains a word that determines its most essential properties, including
  - where it occurs in larger phrases, and
  - what its internal structure is
- This is called the head
- The term “head” is used both for the head word in a phrase and for all the intermediate phrases containing that word
- NB: Not all phrases have heads

Formalizing the Notion of Head

- Expressions have a feature HEAD
- HEAD's values are of type *pos*
- For HEAD values of type *agr-cat*, HEAD's value also includes the feature AGR
- Well-formed trees are subject to the Head Feature Principle

The Head Feature Principle

- Intuitive idea: Key properties of phrases are shared with their heads
- The HFP: In any headed phrase, the HEAD value of the mother and the head daughter must be identical.
- Sometimes described in terms of properties "percolating up" or "filtering down", but this is just metaphorical talk

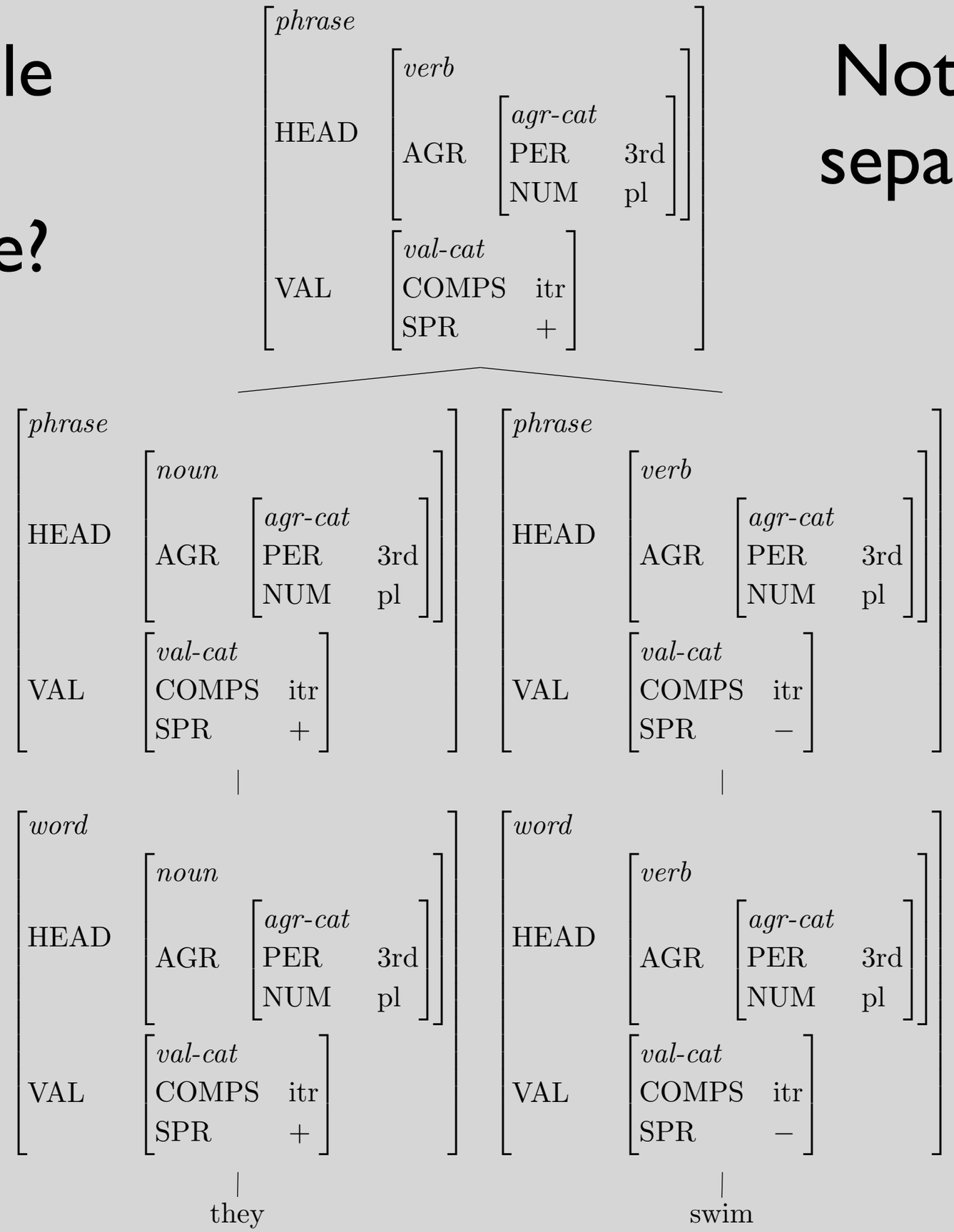
A Tree is Well-Formed if …

- It and each subtree are licensed by a grammar rule or lexical entry
- All general principles (like the HFP) are satisfied.
- NB: Trees are part of our model of the language, so all their features have values (even though we will often be lazy and leave out the values irrelevant to our current point).

Question:

Do phrases that are not headed have HEAD features?

Which rule licenses each node?



Note the three separate uses of DAGs

A Question:

Since the lexical entry for swim below has only [NUM pl] as the value of AGR, how did the tree on the previous slide get [PER 3rd] in the AGR of swim?

$$\left\langle \text{swim} \ , \begin{bmatrix} \textit{word} & & \\ \text{HEAD} & \begin{bmatrix} \textit{verb} & \\ \text{AGR} & \begin{bmatrix} \text{NUM} & \text{pl} \end{bmatrix} \end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & - \end{bmatrix} \end{bmatrix} \right\rangle$$

Overview

- Review: problems with CFG
- Modeling
- Feature structures, unification (pizza)
- Features for linguistic description
- Reformulate grammar rules
- Notion of head/headedness
- Licensing of trees
- Next time: Valence and agreement

Reading Questions

- Why +/- for SPR?
- Why are phrases [COMPS itr], regardless of the COMPS value of the heads inside of them?
- What does this mean? "If a phrase contains its HEAD's complement, it won't combine with any more complements?”
- Why are the nouns all [COMPS itr]? Is this right?
- Why does NOM count as a phrase, even with only one word in it?

Reading Questions

- In section 3.6.3: it states the abbreviation of NOM with [SPR -], I think also there are nominal phrases which have [SPR +] like *Lucy* which doesn't need specifier, should the abbreviation for NOM be more generic and not specify a value for "SPR" ?

- Is writing a lexical entry for a word particular to what sentence this word is in? Since it seems like one word without context can have multiple lexical entries.

Reading Questions

- Is underspecification always intentional?
- What is the point of underspecification?
- What is the value of giving the rules indeterminate HEAD value? This makes sense for COMPS itr, but is there any reason HEAD is not specified as verb for COMPS dtr or str words? Is it just a matter of underspecifying the rule so that the constraint is enforced by the lack of any COMPS dtr or str nouns?

Reading Questions

- Regarding SPR, could you elaborate on how the right-hand side of a rule such as NP→D NOM licenses the left-hand side? For example, D is [SPR +] and NOM is [SPR -]. What exactly ensures that the NP is also [SPR +]?
- Is the feature structure of the mother node the unification of all its daughter nodes?
- How do we handle languages where specifiers are to the right?

Reading Questions

- What work do the feature VAL and the type val-cat do for us? Couldn't we just have flatter feature structures?
- HEAD is used to describe the type of element, and VAL is used to talk about its interactions with other elements. Is that right?

Reading Questions

- What does Head Feature Principle buy to us beside simplifying grammar rules by factoring out common head phrases? For example, can it also be used to validate formulated trees?
- How do we know the HEAD value of the daughter? For example, a word "design" can be either a noun or a verb. How do we build a sentence bottom-up if there exists word-level ambiguity

Reading Questions

- Why are sentences verby?
- Why are NPs more 'complete' than VPs?
- What does 'combinatoric potential' mean?

Reading Questions

- What does [1] stand for in coordination? First guess: same feature structure (like what NP expands to), but you also get *three dogs*, *a cat and some fish*, and those will have different values for NUM.
- Why is there no *adv* under *pos*?

Reading Questions

- Relationships between each branch under the type hierarchy as is shown in (69): Do they exclude each other, or overlap with another, or inherit some features from another? Noun, verb and determiner are under *agr-pos* because three of them take the feature AGR. Then, is *agr-cat* under *feat-strutc* because each sentence or phrase takes this feature? If so, why not include pos under *agr-cat*?

Reading Questions

- I'm wondering about causality in agreement. In *Alex denies the allegation*, both *Alex* and *denies* seem to just happen to get an AGR feature corresponding to 3rd sg - they agree because the grammar says they should agree. However, it seems to me that *denies* gets its respective AGR feature because of the feature that *Alex* has. In other words, *denies* is made to agree with *Alex*, but *Alex* is not really "made" to agree with *denies*. Is there a way to express this sort of causal relationship in this framework?

Reading Questions

- Is this grammar still order-independent?
- What is the relationship between the type hierarchy (with inheritance) and OOP?
- How do we choose which features to include?

Reading Questions

- How much does this buy us in computational accuracy? And how much does it hurt us in computational performance? It does seem quite cumbersome.
- Why do we insist on fully specifying the models?

Reading Questions

- If we use the feature-value system to describe words and phrases, will we have to define a formalism that could basically handle all kinds of foreseeable grammatical scenarios ? How will such formalism handle unstructured text that is prevalent in the social media?

Reading Questions

- In section 3.3.3, the discussion of the linguistic application of feature structures makes mention of mother nodes that share with their "head" daughters. Does this imply that the CFG model enhanced with feature structure representations supports inheritance? How extensive is this concept of mother nodes sharing with their daughters? Could the model be compared to an object oriented programming language like Java that supports paradigms such as abstract classes, inheritance and object instantiation? Could the representation of categories as feature structures be compared to classes and abstract classes while the representation of words as lexical entries be compared to instantiated objects?