Ling/CSE 472: Introduction to Computational Linguistics

5/9/17
Feature structures and unification

Overview

- Problems with CFG
- Feature structures
- Unification
- Agreement
- Subcategorization
- Long-distance Dependencies
- Reading questions

Problems with CFG (with atomic node labels)

- Simple rules, with simple category sets overgenerate:
- What are some non-sentences that this CFG licenses?

S $\rightarrow$	NP VP
NP $\rightarrow$	(Det) Noun
VP $\rightarrow$	Verb (NP) (NP|PP)
PP $\rightarrow$	Prep NP
Noun $\rightarrow$	cat, cats, dog, dogs, I, you, we, they, he, she, it
Det $\rightarrow$	the, a, this, these, some, many
Verb $\rightarrow$	bark, barks, barked, am, is, are, was, were, rely, relies, see, sees, saw
Prep $\rightarrow$	on, in, above, before

Problems with CFG (with atomic node labels)

- How could that be fixed, using the CFG formalism?

$$
\begin{array}{ll}
\text{S} \rightarrow & \text{NP\_sg VP\_sg} \\
\text{S} \rightarrow & \text{NP\_pl VP\_pl} \\
\text{NP\_sg} \rightarrow & \text{(Det\_sg) Noun\_sg} \\
\text{NP\_pl} \rightarrow & \text{(Det\_pl) Noun\_pl} \\
\text{VP\_sg} \rightarrow & \text{V\_intrans\_sg} \\
\text{VP\_sg} \rightarrow & \text{V\_trans\_sg NP\_sg} \\
\text{VP\_sg} \rightarrow & \text{V\_trans\_sg NP\_pl} \\
\ldots &
\end{array}
$$

Generalized Phrase Structure Grammar (GPSG)

- Gazdar et al 1982
- Added feature structures to CFG, but stayed CFG-equivalent
- Eventually, it became generally accepted that natural languages are in fact not context free
- GPSG generalized to HPSG (Pollard & Sag 1994)

Feature Structures

- Break 'atomic' symbols like 'V_intrans_sg' into bundles of information
- Allows for the statement of cross-cutting generalizations

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

Attribute value matrices

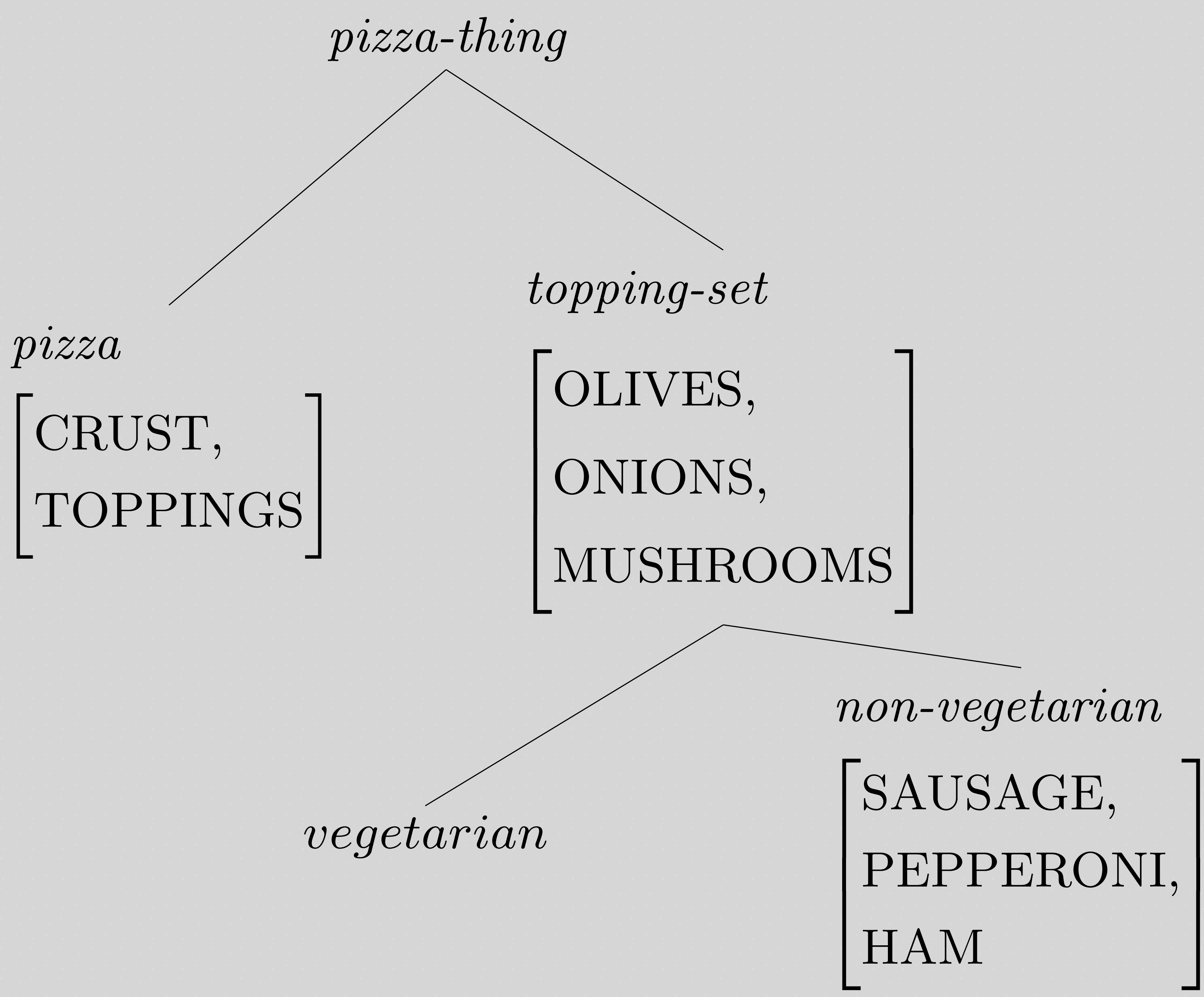
$$\begin{bmatrix} \text{FEATURE}_1 & \text{VALUE}_1 \\ \text{FEATURE}_2 & \text{VALUE}_2 \\ \dots & \\ \text{FEATURE}_n & \text{VALUE}_n \end{bmatrix}$$

- Values can be atomic symbols, or feature structures in their own right.

Unification

- Test whether two feature structures are compatible
- If so, find the most general feature structure that includes all information from both
- Section 15.2 shows unification of untyped feature structures
- Pizza examples (following) add in types (see 15.6)

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

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

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

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

$$= \phi$$

Combining Constraints

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

$$= \phi$$

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

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

Combining Constraints

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

$$=$$

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

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

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

=

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

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

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

$$= \phi$$

Fixing the unwieldy grammar: Agreement

- Awkward CFG analyses

$$
\begin{array}{ll}
\text{S} \rightarrow & \text{NP\_sg VP\_sg} \\
\text{S} \rightarrow & \text{NP\_pl VP\_pl} \\
\text{NP\_sg} \rightarrow & \text{(Det\_sg) Noun\_sg} \\
\text{NP\_pl} \rightarrow & \text{(Det\_pl) Noun\_pl} \\
\text{VP\_sg} \rightarrow & \text{V\_intrans\_sg} \\
\text{VP\_sg} \rightarrow & \text{V\_trans\_sg NP\_sg} \\
\text{VP\_sg} \rightarrow & \text{V\_trans\_sg NP\_pl} \\
\ldots &
\end{array}
$$

Fixing the unwieldy grammar: Agreement

- Better, with unification:

$$
\begin{array}{ll}
S \rightarrow & NP[AGR\ \boxed{1}]\ VP[AGR\ \boxed{1}] \\
NP[AGR\ \boxed{1}] \rightarrow & (Det[AGR\ \boxed{1}])\ Noun[AGR\ \boxed{1}] \\
VP[AGR\ \boxed{1}] \rightarrow & V\_intrans[AGR\ \boxed{1}] \\
VP[AGR\ \boxed{1}] \rightarrow & V\_trans[AGR\ \boxed{1}]\ NP \\
VP[AGR\ \boxed{1}] \rightarrow & V\_pp\_trans[AGR\ \boxed{1}]\ PP \\
VP[AGR\ \boxed{1}] \rightarrow & V\_ditrans[AGR\ \boxed{1}]\ NP\ NP \\
VP[AGR\ \boxed{1}] \rightarrow & V\_pp\_ditrans[AGR\ \boxed{1}]\ NP\ PP \\
\ldots &
\end{array}
$$

Fixing the unwieldy grammar: Subcategorization

$$S \rightarrow \text{NP[AGR } \boxed{1}] \text{ VP[AGR } \boxed{1}]$$
$$\text{NP[AGR } \boxed{1}] \rightarrow (\text{Det[AGR } \boxed{1}]) \text{ Noun[AGR } \boxed{1}]$$
$$\text{VP[AGR } \boxed{1}] \rightarrow \text{V[AGR } \boxed{1}, \text{SUBCAT } \boxed{A}] \boxed{A}$$
$$\text{V[AGR sg, SUBCAT } \langle\ \rangle] \rightarrow \text{sleeps}$$
$$\text{V[AGR pl, SUBCAT } \langle\ \rangle] \rightarrow \text{sleep}$$
$$\text{V[SUBCAT } \langle\ \rangle] \rightarrow \text{slept}$$
$$\text{V[AGR sg, SUBCAT } \langle\ NP \rangle] \rightarrow \text{sees}$$
$$\text{V[AGR pl, SUBCAT } \langle\ NP \rangle] \rightarrow \text{see}$$
$$\text{V[SUBCAT } \langle\ NP \rangle] \rightarrow \text{saw}$$
$$\ldots$$

Examples

- *wh*-questions:

  What did you find?

  Tell me who you talked to

- relative clauses:

  the item that I found

  the guy who(m) I talked to

- topicalization:

  The manual, I can't find

  Chris, you should talk to.

- *easy*-adjectives:

  My house is easy to find.
  Pat is hard to talk to.

What these have in common

- There is a ‘gap’: nothing following *find* and *to*, even though both normally require objects.
- Something that fills the role of the element missing from the gap occurs at the beginning of the clause.
- We use topicalization and *easy*-adjectives to illustrate:

  The manual, I can’t find _____

  Chris is easy to talk to _____

Gaps and their fillers can be far apart:

- *The solution to this problem, Pat said that someone claimed you thought I would never find____.*
- *Chris is easy to consider it impossible for anyone but a genius to try to talk to_____.*

☞ That's why we call them "long distance dependencies"

Fillers often have syntactic properties associated with their gaps

Him, I haven't met___.

**He, I haven't met___.*

The scissors, Pat told us ____ were missing.

**The scissors, Pat told us ____ was missing.*

On Pat, you can rely___.

**To Pat, you can rely___.*

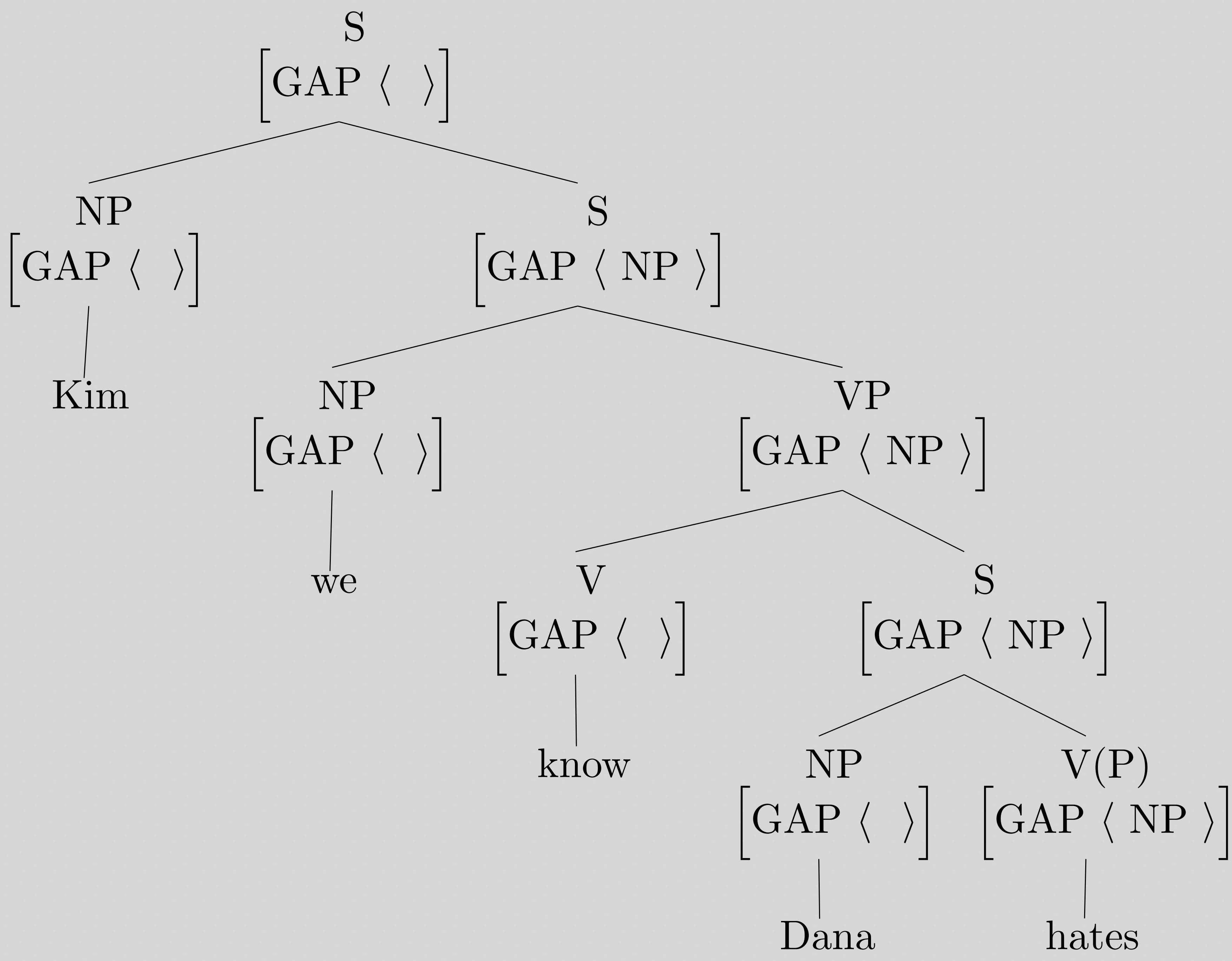
Very Rough Sketch of Our Approach

- A feature GAP records information about a missing constituent.
- The GAP value is passed up the tree by a new principle.
- A new grammar rule expands S as a filler followed by another S whose GAP value matches the filler.
- Caveat: Making the details of this general idea work involves several complications.

A Word with a Non-Empty GAP Value

$$\left\langle \text{hand}\ , \begin{bmatrix} \textit{word} & \\ \text{SYN} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix}\text{FORM} & \text{fin}\end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{SPR} & \langle\ \boxed{1}\ \rangle \\ \text{COMPS} & \langle\ \boxed{3}\text{PP[to]}\ \rangle \end{bmatrix} \\ \text{GAP} & \langle\ \boxed{2}\text{NP[acc]}\ \rangle \end{bmatrix} \\ \text{ARG-ST} & \left\langle \overset{\boxed{1}\text{NP}}{\begin{bmatrix} \text{CASE} & \text{nom} \\ \text{AGR} & \textit{non-3sing}\end{bmatrix}}\ , \boxed{2}\ , \boxed{3} \right\rangle \end{bmatrix} \right\rangle$$

How We Want GAP to Propagate



The Head-Filler Rule

$$[\textit{phrase}] \rightarrow \boxed{1}\begin{bmatrix}\text{GAP} & \langle\ \rangle\end{bmatrix} \quad \mathbf{H}\begin{bmatrix}\text{HEAD} & \begin{bmatrix}\textit{verb} & \\ \text{FORM} & \text{fin}\end{bmatrix} \\ \text{VAL} & \begin{bmatrix}\text{SPR} & \langle\ \rangle \\ \text{COMPS} & \langle\ \rangle\end{bmatrix} \\ \text{STOP-GAP} & \langle\ \boxed{1}\ \rangle \\ \text{GAP} & \langle\ \boxed{1}\ \rangle\end{bmatrix}$$

Overview

- Problems with CFG
- Feature structures
- Unification
- Agreement
- Subcategorization
- Long-distance Dependencies
- Reading questions

Reading questions

- Feature structures and operations greatly resemble set theory to me with operations like unification and the idea of subsuming. Where exactly then, does feature logic and set theory differ?
- What is the application of unification of reentrant structures? Does it mean the merge of features of different grammars?
- So, are reentrant structures just a tool to make AVMs more compact, or is there ever any significance to two features sharing the same node?
- I'm somewhat unclear on the significance of reentrancy. What would be an example of the linguistic use of the sample feature structure at the end of section 15.1?

Reading questions

- What's the base case of the inductive definition of subsumption at the top of page 496?

Subsumption is represented by the operator $\sqsubseteq$. A feature structure F subsumes a feature structure G ($F \sqsubseteq G$) if and only if

1. For every feature x in F, $F(x) \sqsubseteq G(x)$ (where $F(x)$ means "the value of the feature x of feature structure F").
2. For all paths p and q in F such that $F(p) = F(q)$, it is also the case that $G(p) = G(q)$.

Reading questions

- In section 15.2, how does the unification operator not get confused by the [] value, but instead manages to match the value preceding it (in the example: "[NUMBER sg] unified with [NUMBER []] = [NUMBER sg])? Could there be a case in which there are two possibilities, one in which it succeeds, and another in which it fails?

Reading questions

- On pg 504 and 505 there are two different notations for the verb want. Which is more commonly used/which is better? I'm also a little lost on what control information is. (pg 505)

want, might be expressed as

$$
\begin{array}{l}
\textit{Verb} \rightarrow \textit{want} \\
\quad \langle \textit{Verb}\ \text{HEAD SUBCAT FIRST CAT} \rangle = \textit{VP} \\
\quad \langle \textit{Verb}\ \text{HEAD SUBCAT FIRST FORM} \rangle = \textit{infinitive}
\end{array}
$$

$$
\begin{bmatrix}
\text{ORTH} & \textit{want} \\
\text{CAT} & \textit{Verb} \\
\text{HEAD} & \begin{bmatrix} \text{SUBCAT} & \left\langle \begin{bmatrix} \text{CAT } \textit{NP} \end{bmatrix}, \begin{bmatrix} \text{CAT } \textit{VP} \\ \text{HEAD} \begin{bmatrix} \text{VFORM } \textit{infinitival} \end{bmatrix} \end{bmatrix} \right\rangle \end{bmatrix}
\end{bmatrix}
$$

results in a very large s

Reading questions

- Can you describe the property of the syntactic structure of English which allows us to use only the agreement of a Head of the syntactic constituent to determine the agreement of the whole?

- Verbs like "serve" work for multiple agreement cases: first and second person, and third person plural (I/we/you/they serve). Would the agreement structure include this extra information, or is the information stored only relevant to the specific sentence?

Reading questions

- What's the relationship between feature structures and disambiguation?
- How deep down the rabbit-hole of semantic information do feature structures go? It's mentioned that they encode basic semantic things, like count / mass, but do they encode more complicated things?

Reading questions

- Is the addition of constraints to CFGs kind of like "extended" regular expressions (which made them not regular), or does the addition of constraints not give any actual power? If so, are CFGs + constraints less, equally, or more powerful than context-sensitive grammars?
- Feature paths look like FSMs... Are we going to use them again for feature structures or is it coincidental?
- How are feature structures read and implemented by a machine? It seems like they are too dense to be used simultaneously or in conjunction with a parser.

Reading questions

- Could feature structure be applied on the structure of the whole sentence? For example, we could look at the feature of the surrounding words to determine the meaning of "bank" in a give sentence.

- Are feature structures like this ever used for things like theta-roles, which encode information which is more semantic but still relevant to syntax. If they aren't, how are the grammar rules prevented from overgenerating sentences like "the train ate my desk"?

- Also, are feature structures ever used in applications outside of syntax/semantics, like phonology?