Feature Structures, Unification

Scott Farrar
CLMA, University of Washington
farrar@u.washington.edu

February 3, 2010

Today's lecture

1. Some grammatical phenomena
   - Linguistic features
2. Feature structures
3. Operations on feature structures
   - Subsumption
   - Unification
4. Features in the NLTK
   - Creating feature structures
   - FeatStruct behavior

Verb types

Consider the phrase structure rules in the PTB for verbs:

- $VP \rightarrow VBZ\ NP$
  ...breaks the jar
- $VP \rightarrow VBP\ NP$
  ...break the jar
- $VP \rightarrow VBD\ NP$
  ...broke the jar
- $VP \rightarrow VBN\ NP$
  ...broken the jar

Verb types

Consider the phrase structure rules in the PTB for verbs:

- $VP \rightarrow VBZ\ NP$
  ...breaks the jar
- $VP \rightarrow VBP\ NP$
  ...break the jar
- $VP \rightarrow VBD\ NP$
  ...broke the jar
- $VP \rightarrow VBN\ NP$
  ...broken the jar

We could capture the regularity by a rule like this:
$VP \rightarrow VBx\ NP$
What exactly is ‘x’ representing?

Noun types

And for nouns:

- *NP* → *DT NN*
  ...the book
- *NP* → *DT NNS*
  ...the books

Noun types

And for nouns:

- $NP \rightarrow DT\ NN$
  ...the book
- $NP \rightarrow DT\ NNS$
  ...the books

We could capture the regularity by a rule like this:
$NP \rightarrow DT\ NNx$
What exactly is 'x' representing?

Subject-Verb Agreement

- *The parrot talk*<u>*s*</u>*.*

Subject-Verb Agreement

- *The parrot talk<u>s</u>.*
- **The parrot talk.*

Subject-Verb Agreement

- *The parrot talk*<u>*s*</u>*.*
- **The parrot talk.*
- *They know Mary.*

Subject-Verb Agreement

- *The parrot talks.*
- **The parrot talk.*
- *They know Mary.*
- **They knows Mary.*

Subject-Verb Agreement

- *The parrot talks.*
- **The parrot talk.*
- *They know Mary.*
- **They knows Mary.*
- *We sing.*

Subject-Verb Agreement

- *The parrot talks.*
- **The parrot talk.*
- *They know Mary.*
- **They knows Mary.*
- *We sing.*
- **We sings.*

Subject-Verb Agreement

- *The parrot talks.*
- **The parrot talk.*
- *They know Mary.*
- **They knows Mary.*
- *We sing.*
- **We sings.*

Furthermore agreement infomation is not captured in treebanks such as the PTB:
$S \rightarrow NP\ VP$
The *NNx* in the *NP* does not depend on the *VBx* in the *VP*.

Features and values

Definition

linguistic feature: a property-like element that changes the grammatical behavior of syntactic constituents; the elements into which linguistic units, such as words, can be broken down. For a given sub-domain of grammar, there is a relevant feature:

Features and values

Definition

linguistic feature: a property-like element that changes the grammatical behavior of syntactic constituents; the elements into which linguistic units, such as words, can be broken down. For a given sub-domain of grammar, there is a relevant feature:

- **person**: *I go, you go, he goes*

Features and values

Definition

linguistic feature: a property-like element that changes the grammatical behavior of syntactic constituents; the elements into which linguistic units, such as words, can be broken down. For a given sub-domain of grammar, there is a relevant feature:

- **person**: *I go, you go, he goes*
- **number**: *he dances, they dance*

Features and values

Definition

linguistic feature: a property-like element that changes the grammatical behavior of syntactic constituents; the elements into which linguistic units, such as words, can be broken down. For a given sub-domain of grammar, there is a relevant feature:

- **person**: *I go, you go, he goes*
- **number**: *he dances, they dance*
- **case**: *he brings John, John brings him*

Features and values

Definition

linguistic feature: a property-like element that changes the grammatical behavior of syntactic constituents; the elements into which linguistic units, such as words, can be broken down. For a given sub-domain of grammar, there is a relevant feature:

- **person**: *I go, you go, he goes*
- **number**: *he dances, they dance*
- **case**: *he brings John, John brings him*
- **tense**: *go, went, gone*

Features and values

Definition

linguistic feature: a property-like element that changes the grammatical behavior of syntactic constituents; the elements into which linguistic units, such as words, can be broken down. For a given sub-domain of grammar, there is a relevant feature:

- **person**: *I go, you go, he goes*
- **number**: *he dances, they dance*
- **case**: *he brings John, John brings him*
- **tense**: *go, went, gone*
- **modality**: *may, can,*

Features and values

Definition

linguistic feature: a property-like element that changes the grammatical behavior of syntactic constituents; the elements into which linguistic units, such as words, can be broken down. For a given sub-domain of grammar, there is a relevant feature:

- **person**: *I go, you go, he goes*
- **number**: *he dances, they dance*
- **case**: *he brings John, John brings him*
- **tense**: *go, went, gone*
- **modality**: *may, can,*
- **honorifics** (e.g., Japanese)

Features and values

Definition

linguistic feature: a property-like element that changes the grammatical behavior of syntactic constituents; the elements into which linguistic units, such as words, can be broken down. For a given sub-domain of grammar, there is a relevant feature:

- **person**: *I go, you go, he goes*
- **number**: *he dances, they dance*
- **case**: *he brings John, John brings him*
- **tense**: *go, went, gone*
- **modality**: *may, can,*
- **honorifics** (e.g., Japanese)
- **evidentiality** (e.g., Shipibo)

Features and values

Definition

linguistic feature: a property-like element that changes the grammatical behavior of syntactic constituents; the elements into which linguistic units, such as words, can be broken down. For a given sub-domain of grammar, there is a relevant feature:

- **person**: *I go, you go, he goes*
- **number**: *he dances, they dance*
- **case**: *he brings John, John brings him*
- **tense**: *go, went, gone*
- **modality**: *may, can,*
- **honorifics** (e.g., Japanese)
- **evidentiality** (e.g., Shipibo)
- **noun class** (e.g., Chinese)

Features and values

Definition

Features are usually modeled as pairs of feature **names** and **feature values**. A **value** refers to the language-specific phenomenon within the sub-domain:

Features and values

Definition

Features are usually modeled as pairs of feature **names** and **feature values**. A **value** refers to the language-specific phenomenon within the sub-domain:

- **person**: 1st, 2nd, 3rd

Features and values

Definition

Features are usually modeled as pairs of feature **names** and **feature values**. A **value** refers to the language-specific phenomenon within the sub-domain:

- **person**: 1st, 2nd, 3rd
- **number**: singular, plural, paucal, dual, trial, etc.

Features and values

Definition

Features are usually modeled as pairs of feature **names** and **feature values**. A **value** refers to the language-specific phenomenon within the sub-domain:

- **person**: 1st, 2nd, 3rd
- **number**: singular, plural, paucal, dual, trial, etc.
- **case**: accusative, ergative, locative, directional

Features and values

Definition

Features are usually modeled as pairs of feature **names** and **feature values**. A **value** refers to the language-specific phenomenon within the sub-domain:

- **person**: 1st, 2nd, 3rd
- **number**: singular, plural, paucal, dual, trial, etc.
- **case**: accusative, ergative, locative, directional
- **tense**: past, present, future, hodiernal past, hesternal past

Features and values

Definition

Features are usually modeled as pairs of feature **names** and **feature values**. A **value** refers to the language-specific phenomenon within the sub-domain:

- **person**: 1st, 2nd, 3rd
- **number**: singular, plural, paucal, dual, trial, etc.
- **case**: accusative, ergative, locative, directional
- **tense**: past, present, future, hodiernal past, hesternal past
- **modality**: conditional, subjunctive, abilitative

Features and values

Definition

Features are usually modeled as pairs of feature **names** and **feature values**. A **value** refers to the language-specific phenomenon within the sub-domain:

- **person**: 1st, 2nd, 3rd
- **number**: singular, plural, paucal, dual, trial, etc.
- **case**: accusative, ergative, locative, directional
- **tense**: past, present, future, hodiernal past, hesternal past
- **modality**: conditional, subjunctive, abilitative
- **honorifics**: plus honorific, minus honorific

Features and values

Definition

Features are usually modeled as pairs of feature **names** and **feature values**. A **value** refers to the language-specific phenomenon within the sub-domain:

- **person**: 1st, 2nd, 3rd
- **number**: singular, plural, paucal, dual, trial, etc.
- **case**: accusative, ergative, locative, directional
- **tense**: past, present, future, hodiernal past, hesternal past
- **modality**: conditional, subjunctive, abilitative
- **honorifics**: plus honorific, minus honorific
- **evidentiality**: visual, auditory, hearsay, folklore

Features and values

Definition

Features are usually modeled as pairs of feature **names** and **feature values**. A **value** refers to the language-specific phenomenon within the sub-domain:

- **person**: 1st, 2nd, 3rd
- **number**: singular, plural, paucal, dual, trial, etc.
- **case**: accusative, ergative, locative, directional
- **tense**: past, present, future, hodiernal past, hesternal past
- **modality**: conditional, subjunctive, abilitative
- **honorifics**: plus honorific, minus honorific
- **evidentiality**: visual, auditory, hearsay, folklore
- **noun class**: I, II, III, etc.

Features and values

Definition

Features are usually modeled as pairs of feature **names** and **feature values**. A **value** refers to the language-specific phenomenon within the sub-domain:

- **person**: 1st, 2nd, 3rd
- **number**: singular, plural, paucal, dual, trial, etc.
- **case**: accusative, ergative, locative, directional
- **tense**: past, present, future, hodiernal past, hesternal past
- **modality**: conditional, subjunctive, abilitative
- **honorifics**: plus honorific, minus honorific
- **evidentiality**: visual, auditory, hearsay, folklore
- **noun class**: I, II, III, etc.

See http://grammaticalfeatures.net for a nice summary of features from a strictly linguistics perspective.

Ontology of features

Whereas NPs, VPs, NNs, etc. are conceived of as categories (first-order “citizens”) in the grammar, features are more **property-like**.

Ontology of features

Whereas NPs, VPs, NNs, etc. are conceived of as categories (first-order "citizens") in the grammar, features are more **property-like**.

- The VP has the feature value 'past tense'
- The verb is a 'past tense' verb
- The noun has a case feature 'absolutive'.

Ontology of features

Whereas NPs, VPs, NNs, etc. are conceived of as categories (first-order "citizens") in the grammar, features are more **property-like**.

- The VP has the feature value 'past tense'
- The verb is a 'past tense' verb
- The noun has a case feature 'absolutive'.

Just like, for example, an object like a **car** can have a color property **red**.

Bundles of feature (values)

Sometimes features are conceived of as the atomic units that compose more complex categories.

Bundles of feature (values)

Sometimes features are conceived of as the atomic units that compose more complex categories.

A noun is a **feature bundle** of semantic, morphological and phonological features:

$$\begin{bmatrix} \text{form} & \text{`dog'} \\ \text{number} & \text{singular} \\ \text{animacy} & \text{animate} \end{bmatrix}$$

Today's lecture

1. Some grammatical phenomena
   - Linguistic features
2. Feature structures
3. Operations on feature structures
   - Subsumption
   - Unification
4. Features in the NLTK
   - Creating feature structures
   - FeatStruct behavior

Feature structures

A feature structure is, informally speaking, a set of feature names and values:

$$\begin{bmatrix} \text{feature}_1 & \text{value}_1 \\ \text{feature}_2 & \text{value}_2 \\ \text{feature}_3 & \text{value}_3 \end{bmatrix}$$

A feature structure is defined according to particular grammatical traditions:

- Functional Unification Grammar
- Lexical Functional Grammar
- Head-Driven Phrase Structure Grammar (Ling566)

Typed feature structures: HPSG formal definition

Definition

A typed feature structure is defined on a finite set of features *Feat* and a type hierarchy $\langle Type, \sqsubseteq \rangle$. It is a tuple $\langle Q, r, \delta, \theta \rangle$ where:

- Q is a finite set of nodes,
- $r \in Q$ (r is the root node)
- $\theta : Q \longrightarrow Type$ is a partial typing function
- $\delta : Q \times Feat \longrightarrow Q$ is a partial feature value function

Typed feature structures: HPSG formal definition

Definition

A typed feature structure is defined on a finite set of features *Feat* and a type hierarchy $\langle Type, \sqsubseteq \rangle$. It is a tuple $\langle Q, r, \delta, \theta \rangle$ where:

- Q is a finite set of nodes,
- $r \in Q$ (r is the root node)
- $\theta : Q \longrightarrow Type$ is a partial typing function
- $\delta : Q \times Feat \longrightarrow Q$ is a partial feature value function

subject to the following conditions:

1. r isn't a θ-descendant.
2. all members of Q except r are θ-descendants of r.
3. Some systems add an extra (no cycles) condition: there is no node n or path π such that $\delta(n, \pi) = n$.

from Copestake (2000), Appendix.

More about feature values

atomic value: an unstructured value, one with only one part

$$\begin{bmatrix} \text{tense} & \textbf{past} \\ \text{person} & \textbf{2} \end{bmatrix}$$

More about feature values

atomic value: an unstructured value, one with only one part

$$\begin{bmatrix} \text{tense} & \textbf{past} \\ \text{person} & \mathbf{2} \end{bmatrix}$$

complex value: a structured value, itself a feature structure

$$\begin{bmatrix} \text{tense} & \text{past} \\ \text{agreement} & \begin{bmatrix} \textbf{person} & \mathbf{2} \\ \textbf{number} & \textbf{singular} \end{bmatrix} \end{bmatrix}$$

Graphs or matrices

Feature structures are a type of attribute-value matrix (AVM), a more generalized data structure used to represent all kinds of information. Feature structures (and AVMs) can be represented as graphs with nodes and arcs (cf. text book figures)

Graphs or matrices

Feature structures are a type of attribute-value matrix (AVM), a more generalized data structure used to represent all kinds of information. Feature structures (and AVMs) can be represented as graphs with nodes and arcs (cf. text book figures)

- **Nodes** correspond to variable values and the paths to the variable names.
- **Arcs** represent the feature names.
- A **feature path** is a list of features through a feature structure leading to a particular value.

Feature path

$$\begin{bmatrix} \text{tense} & \text{past} \\ \text{agreement} & \begin{bmatrix} \text{person} & 2 \\ \text{number} & \text{singular} \end{bmatrix} \end{bmatrix}$$

Path:
$\langle \textit{agreement number} \rangle = \text{singular}$

Feature path

$$\begin{bmatrix} \text{tense} & \text{past} \\ \textbf{agreement} & \begin{bmatrix} \text{person} & 2 \\ \text{number} & \text{singular} \end{bmatrix} \end{bmatrix}$$

Path:

$\langle$**agreement** *number*$\rangle$ = singular

Feature path

$$\begin{bmatrix} \text{tense} & \text{past} \\ \textbf{agreement} & \begin{bmatrix} \text{person} & 2 \\ \textbf{number} & \text{singular} \end{bmatrix} \end{bmatrix}$$

Path:
$\langle \textbf{agreement number} \rangle = \text{singular}$

Feature path

$$\begin{bmatrix} \text{tense} & \text{past} \\ \textbf{agreement} & \begin{bmatrix} \text{person} & 2 \\ \textbf{number} & \textbf{singular} \end{bmatrix} \end{bmatrix}$$

Path:

$\langle$**agreement number**$\rangle =$ **singular**

Reentrant structures

Definition

Reentrant structure: one in which the attribute has a value that is another feature structure.

$$\begin{bmatrix} \text{S} & \begin{bmatrix} \text{head} & \boxed{1} \end{bmatrix} \\ \text{NP} & \begin{bmatrix} \text{head} & \begin{bmatrix} \text{agr} & \boxed{2} \end{bmatrix} \end{bmatrix} \\ \text{VP} & \begin{bmatrix} \text{head} & \boxed{1} \begin{bmatrix} \text{agr} & \boxed{2} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Reentrant structures

Definition

Reentrant structure: one in which the attribute has a value that is another feature structure.

$$\begin{bmatrix} \text{S} & \begin{bmatrix} \text{head} & \boxed{1} \end{bmatrix} \\ \text{NP} & \begin{bmatrix} \text{head} & \begin{bmatrix} \text{agr} & \boxed{2} \end{bmatrix} \end{bmatrix} \\ \text{VP} & \begin{bmatrix} \text{head} & \boxed{1} \begin{bmatrix} \text{agr} & \boxed{2} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

A feature structure can be used to represent partial information.

More linguistic examples

Feature structure for a verb:

$$\begin{bmatrix} \text{cat} & \text{VB} \\ \text{lex} & |\,\mathit{GO}\,| \\ \text{tense} & \text{past} \\ \text{aspect} & \text{progressive} \\ \text{form} & \text{``was going''} \end{bmatrix}$$

Feature structure for a noun:

$$\begin{bmatrix} \text{cat} & \text{NN} \\ \text{lex} & |\,\mathit{dog}\,| \\ \text{number} & \text{PL} \\ \text{form} & \text{``dogs''} \end{bmatrix}$$

Linguistic examples

Feature structure for a noun phrase (e.g., *the boy*):

$$\begin{bmatrix} \text{cat} & \text{NP} \\ \text{spec} & \boxed{1}\begin{bmatrix} \text{cat} & \text{DT} \\ \text{number} & \text{SG} \\ \text{definite} & \text{yes} \\ \text{form} & \text{``the''} \end{bmatrix} \\ \text{head} & \boxed{2}\begin{bmatrix} \text{cat} & \text{NN} \\ \text{number} & \text{SG} \\ \text{lex} & |\ \textit{boy}\ | \\ \text{form} & \text{``boy''} \end{bmatrix} \\ \text{definite} & \text{yes} \\ \text{number} & \text{SG} \end{bmatrix}$$

Today's lecture

1. Some grammatical phenomena
   - Linguistic features
2. Feature structures
3. Operations on feature structures
   - Subsumption
   - Unification
4. Features in the NLTK
   - Creating feature structures
   - FeatStruct behavior

Subsumption

Definition

Subsumption is a relation that holds between feature structures. A less specific (more abstract) feature structure subsumes an equally or more specific one. The subsumption symbol is $\sqsubseteq$.

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix}$$

Subsumption

Definition

Subsumption is a relation that holds between feature structures. A less specific (more abstract) feature structure subsumes an equally or more specific one. The subsumption symbol is $\sqsubseteq$.

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \sqsubseteq \begin{bmatrix} \text{number} & \text{PL} \\ \text{person} & 3 \end{bmatrix}$$

Subsumption

Definition

Subsumption is a relation that holds between feature structures. A less specific (more abstract) feature structure subsumes an equally or more specific one. The subsumption symbol is $\sqsubseteq$.

$$\begin{bmatrix}\text{number} & \text{PL}\end{bmatrix} \sqsubseteq \begin{bmatrix}\text{number} & \text{PL}\\ \text{person} & 3\end{bmatrix} \sqsubseteq \begin{bmatrix}\text{number} & \text{PL}\\ \text{person} & 3\\ \text{lex} & \left| \textit{dog} \right|\end{bmatrix}$$

Subsumption

$$\begin{bmatrix} \text{number} & \text{PL} \\ \text{person} & 3 \end{bmatrix} \qquad\qquad \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix}$$

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \quad ? \qquad\qquad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

$$\begin{bmatrix} \quad \end{bmatrix} \quad ? \qquad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

Subsumption

$$\begin{bmatrix} \text{number} & \text{PL} \\ \text{person} & 3 \end{bmatrix} \quad ? \qquad \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix}$$

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \quad ? \qquad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

$$\begin{bmatrix} \quad \end{bmatrix} \quad ? \qquad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

Subsumption

$$\begin{bmatrix} \text{number} & \text{PL} \\ \text{person} & 3 \end{bmatrix} \quad \neg \quad \sqsubseteq \quad \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix}$$

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \quad ? \qquad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

$$\begin{bmatrix} \; \end{bmatrix} \quad ? \qquad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

Subsumption

$$\begin{bmatrix} \text{number} & \text{PL} \\ \text{person} & 3 \end{bmatrix} \quad \neg \quad \sqsubseteq \quad \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix}$$

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \quad ? \qquad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

$$\begin{bmatrix} \quad \end{bmatrix} \quad ? \qquad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

Subsumption

$$\begin{bmatrix} \text{number} & \text{PL} \\ \text{person} & 3 \end{bmatrix} \quad \neg \sqsubseteq \quad \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix}$$

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \quad \neg \sqsubseteq \quad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

$$\begin{bmatrix} \; \end{bmatrix} \quad ? \quad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

Subsumption

$$\begin{bmatrix} \text{number} & \text{PL} \\ \text{person} & 3 \end{bmatrix} \quad \neg \quad \sqsubseteq \quad \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix}$$

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \quad \neg \quad \sqsubseteq \quad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

$$\begin{bmatrix} \quad \end{bmatrix} \quad ? \quad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

Subsumption

$$\begin{bmatrix} \text{number} & \text{PL} \\ \text{person} & 3 \end{bmatrix} \quad \neg \quad \sqsubseteq \quad \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix}$$

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \quad \neg \quad \sqsubseteq \quad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

$$\begin{bmatrix} \quad \end{bmatrix} \quad \sqsubseteq \quad \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix}$$

Subsumption

Criteria for $\sqsubseteq$

$F \sqsubseteq G$ iff:

- 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".
- For all paths p and q in F such that $F(p) = F(q)$, it is also the case that $G(p) = G(q)$.

The unification operation

Definition

Unification is a binary operation over two features structures $f1$ and $f2$, used for comparing or combining information.

The unification operation

Definition

Unification is a binary operation over two features structures $f1$ and $f2$, used for comparing or combining information.

Unification of $f1$ and $f2$ either returns a merged feature structure with the information from both $f1$ and $f2$, or false if $f1$ and $f2$ are incompatible. The unification operator is represented by: $\sqcup$.

The unification operation

Definition

Unification is a binary operation over two features structures $f1$ and $f2$, used for comparing or combining information.

Unification of $f1$ and $f2$ either returns a merged feature structure with the information from both $f1$ and $f2$, or false if $f1$ and $f2$ are incompatible. The unification operator is represented by: $\sqcup$.

$$\begin{bmatrix}\text{number} & \text{PL}\end{bmatrix} \sqcup \begin{bmatrix}\text{person} & 2\end{bmatrix}$$

The unification operation

Definition

Unification is a binary operation over two features structures $f1$ and $f2$, used for comparing or combining information.

Unification of $f1$ and $f2$ either returns a merged feature structure with the information from both $f1$ and $f2$, or false if $f1$ and $f2$ are incompatible. The unification operator is represented by: $\sqcup$.

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \sqcup \begin{bmatrix} \text{person} & 2 \end{bmatrix} = \begin{bmatrix} \text{number} & \text{PL} \\ \text{person} & 2 \end{bmatrix}$$

The unification operation

For comparing information in two feature structures, unification can be used. Are two feature structures the same, or are they incompatible?

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \sqcup \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} =$$

The unification operation

For comparing information in two feature structures, unification can be used. Are two feature structures the same, or are they incompatible?

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \sqcup \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} = \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix}$$

The unification operation

For comparing information in two feature structures, unification can be used. Are two feature structures the same, or are they incompatible?

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \sqcup \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} = \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix}$$

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \sqcup \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix} =$$

The unification operation

For comparing information in two feature structures, unification can be used. Are two feature structures the same, or are they incompatible?

$$\begin{bmatrix}\text{number} & \text{PL}\end{bmatrix} \sqcup \begin{bmatrix}\text{number} & \text{PL}\end{bmatrix} = \begin{bmatrix}\text{number} & \text{PL}\end{bmatrix}$$

$$\begin{bmatrix}\text{number} & \text{PL}\end{bmatrix} \sqcup \begin{bmatrix}\text{number} & \text{SG}\end{bmatrix} = \text{False}$$

The unification operation

For comparing information in two feature structures, unification can be used. Are two feature structures the same, or are they incompatible?

$$\begin{bmatrix}\text{number} & \text{PL}\end{bmatrix} \sqcup \begin{bmatrix}\text{number} & \text{PL}\end{bmatrix} = \begin{bmatrix}\text{number} & \text{PL}\end{bmatrix}$$

$$\begin{bmatrix}\text{number} & \text{PL}\end{bmatrix} \sqcup \begin{bmatrix}\text{number} & \text{SG}\end{bmatrix} = \text{False}$$

Unification preserves and possibly adds information to the resulting feature structure. This property is called:

The unification operation

For comparing information in two feature structures, unification can be used. Are two feature structures the same, or are they incompatible?

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \sqcup \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} = \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix}$$

$$\begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \sqcup \begin{bmatrix} \text{number} & \text{SG} \end{bmatrix} = \text{False}$$

Unification preserves and possibly adds information to the resulting feature structure. This property is called: **monotonicity**

Unification of dissimilar structures

$$\begin{bmatrix}\text{agreement} & \begin{bmatrix}\text{number} & \text{PL}\end{bmatrix}\end{bmatrix} \sqcup \begin{bmatrix}\text{number} & \text{PL}\end{bmatrix} =$$

Unification of dissimilar structures

$$\begin{bmatrix} \text{agreement} & \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \end{bmatrix} \sqcup \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} = \begin{bmatrix} \text{agreement} & \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} \\ \begin{bmatrix} \text{number} & \text{PL} \end{bmatrix} & \end{bmatrix}$$

Unification: example

Grammar entry for a sentence

$$\begin{bmatrix} \text{cat} & \text{S} \\ \text{voice} & \text{active} \\ \text{agent} & \boxed{1}\begin{bmatrix} \text{cat} & \text{NP} \\ \text{number} & \boxed{4} \end{bmatrix} \\ \text{process} & \boxed{2}\begin{bmatrix} \text{cat} & \text{VB} \\ \text{number} & \boxed{4} \end{bmatrix} \\ \text{patient} & \boxed{3}\begin{bmatrix} \text{cat} & \text{NP} \end{bmatrix} \\ \text{pattern} & \begin{bmatrix} \text{subject} & \boxed{1} \\ \text{verb} & \boxed{2} \\ \text{object} & \boxed{3} \end{bmatrix} \end{bmatrix}$$

Unification: example

Grammar entry for a noun phrase

$$\begin{bmatrix} \text{cat} & \text{NP} \\ \text{spec} & \boxed{1}\begin{bmatrix}\text{cat} & \text{DT}\\ \text{number} & \boxed{3} \\ \text{definite} & \boxed{4}\end{bmatrix} \\ \text{head} & \boxed{2}\begin{bmatrix}\text{cat} & \text{NN}\\ \text{number} & \boxed{3}\end{bmatrix} \\ \text{number} & \boxed{3} \\ \text{definite} & \boxed{4} \\ \text{pattern} & \begin{bmatrix}\text{first} & \boxed{1}\\ \text{second} & \boxed{2}\end{bmatrix}\end{bmatrix}$$

Unification: example

Lexical entries in AVM form

$$\begin{bmatrix} \text{cat} & \text{DT} \\ \text{definite} & \text{yes} \\ \text{number} & \text{SG} \\ \text{form} & \text{“the”} \end{bmatrix}$$

$$\begin{bmatrix} \text{cat} & \text{DT} \\ \text{definite} & \text{yes} \\ \text{number} & \text{PL} \\ \text{form} & \text{“these”} \end{bmatrix}$$

Unification: example

Unifying a noun phrase with a determiner

$$\begin{bmatrix} \text{cat} & \text{NP} \\ \text{spec} & \boxed{1}\begin{bmatrix} \text{cat} & \text{DT} \\ \text{number} & \boxed{3} \\ \text{definite} & \boxed{4} \end{bmatrix} \\ \text{head} & \boxed{2}\begin{bmatrix} \text{cat} & \text{NN} \\ \text{number} & \boxed{3} \end{bmatrix} \\ \text{number} & \boxed{3} \\ \text{definite} & \boxed{4} \\ \text{pattern} & \begin{bmatrix} \text{first} & \boxed{1} \\ \text{second} & \boxed{2} \end{bmatrix} \end{bmatrix} \sqcup \begin{bmatrix} \text{cat} & \text{DT} \\ \text{definite} & \text{yes} \\ \text{number} & \text{PL} \\ \text{form} & \text{``these''} \end{bmatrix}$$

Unification: example

Unifying the right part of a noun phrase with a determiner

$$\begin{bmatrix} \text{cat} & \text{DT} \\ \text{number} & \boxed{3} \\ \text{definite} & \boxed{4} \end{bmatrix} \sqcup \begin{bmatrix} \text{cat} & \text{DT} \\ \text{definite} & \text{yes} \\ \text{number} & \text{PL} \\ \text{form} & \text{``these''} \end{bmatrix} =$$

Unification: example

Unifying the right part of a noun phrase with a determiner

$$\begin{bmatrix} \text{cat} & \text{DT} \\ \text{number} & \boxed{3} \\ \text{definite} & \boxed{4} \end{bmatrix} \sqcup \begin{bmatrix} \text{cat} & \text{DT} \\ \text{definite} & \text{yes} \\ \text{number} & \text{PL} \\ \text{form} & \text{``these''} \end{bmatrix} = \begin{bmatrix} \text{cat} & \text{DT} \\ \text{definite} & \text{yes} \\ \text{number} & \text{PL} \\ \text{form} & \text{``these''} \end{bmatrix}$$

Unification: example

Result of unification

$$\begin{bmatrix} \text{cat} & \text{NP} \\ \text{spec} & \boxed{1}\begin{bmatrix} \text{cat} & \text{DT} \\ \text{number} & \text{PL} \\ \text{definite} & \text{yes} \\ \text{form} & \text{``these''} \end{bmatrix} \\ \text{head} & \boxed{2}\begin{bmatrix} \text{cat} & \text{NN} \\ \text{number} & \text{PL} \end{bmatrix} \\ \text{number} & \text{PL} \\ \text{definite} & \text{yes} \\ \text{pattern} & \begin{bmatrix} \text{first} & \boxed{1} \\ \text{second} & \boxed{2} \end{bmatrix} \end{bmatrix}$$

Today's lecture

1. Some grammatical phenomena
    - Linguistic features
2. Feature structures
3. Operations on feature structures
    - Subsumption
    - Unification
4. Features in the NLTK
    - Creating feature structures
    - FeatStruct behavior

The main features module

The NLTK has a module for feature structures and accompanying operations: `nltk.featstruct`

The basic feature structure `nltk.featstruct.FeatStruct` is implemented as a mapping (dictionary) from features to values.

```
>>>from nltk.featstruct import FeatStruct

>>>fs1 = FeatStruct(number=‘singular’, person=3)

>>>print fs1

[ number = ‘singular’ ]
[ person = 3          ]
```

The main features module

```
>>>fs2 = FeatStruct(type=‘NP’, agr=fs1)

>>>print fs2


[ agr  = [ number = ‘singular’ ] ]
[        [ person = 3          ] ]
[                                ]
[ type = ’NP’                    ]
```

Unification in the NLTK

The NLTK has an implementation of unification with `nltk.featstruct.FeatStruct.unify()`.

```
fs1

[ agr  = [ number = ‘singular’ ] ]
[        [ person = 3          ] ]
[                                ]
[ type = ’NP’                    ]

fs2

[ agr  = [ number = ?n ] ]
[                        ]
[ subj = [ number = ?n ] ]
```

Unification in the NLTK

```
>>>print fs2.unify(fs3)
[ agr  = [ number = ‘singular’ ] ]
[        [ person = 3          ] ]
[                                ]
[ subj = [ number = ‘singular’ ] ]
[                                ]
[ type = ‘NP’                    ]
```

Creating feature structures: FeatStruct

```
fs1 = FeatStruct(agreement= \
        FeatStruct(number=‘singular’, person=‘third’))
```

Creating feature structures: FeatStruct

```
fs1 = FeatStruct(agreement= \
        FeatStruct(number=‘singular’, person=‘third’))

[ agreement = [ number = ‘singular’ ] ]
[             [ person = ‘third’    ] ]
```

Creating feature structures: FeatStruct

```
fs1 = FeatStruct(agreement= \
        FeatStruct(number=‘singular’, person=‘third’))
```

```
[ agreement = [ number = ‘singular’ ] ]
[             [ person = ‘third’    ] ]
```

Use the `FeatStruct` constructor; note how quotes are used.

Creating feature structures, short-hand

```
fs1 = FeatStruct("[agreement = \
        [number=‘singular’,person=‘third’]]")
```

Creating feature structures, short-hand

```
fs1 = FeatStruct("[agreement = \
        [number=‘singular’,person=‘third’]]")

[ agreement = [ number = ‘singular’ ] ]
[             [ person = ‘third’    ] ]
```

Creating feature structures, short-hand

```
fs1 = FeatStruct("[agreement = \
        [number=‘singular’,person=‘third’]]")
```

```
[ agreement = [ number = ‘singular’ ] ]
[             [ person = ‘third’    ] ]
```

The feature structure is parsed from the string. Values are quoted.

Creating feature structures, short-hand w/o value quotes

```
fs1 = FeatStruct("[agreement = \
       [number=singular,person=third]]")
```

Creating feature structures, short-hand w/o value quotes

```
 fs1 = FeatStruct("[agreement = \
        [number=singular,person=third]]")

[ agreement = [ number = ‘singular’ ] ]
[             [ person = ‘third’    ] ]
```

Creating feature structures, short-hand w/o value quotes

```
 fs1 = FeatStruct("[agreement = \
        [number=singular,person=third]]")
```

```
[ agreement = [ number = ‘singular’ ] ]
[             [ person = ‘third’    ] ]
```

Value quotes aren’t necessary.

Using indices

```
fs = FeatStruct("[\
     NAME=Lee, \
     ADDRESS=(1)[NUMBER=74, STREET='rue Pascal'],\
     SPOUSE=[NAME=Kim, ADDRESS->(1)],]")
```

Using indices

```
fs = FeatStruct("[\
     NAME=Lee, \
     ADDRESS=(1)[NUMBER=74, STREET=‘rue Pascal’],\
     SPOUSE=[NAME=Kim, ADDRESS->(1)],]")
```

```
[ ADDRESS = (1) [ NUMBER = 74           ] ]
[               [ STREET = ‘rue Pascal’ ] ]
[                                         ]
[ NAME    = ’Lee’                         ]
[                                         ]
[ SPOUSE  = [ ADDRESS -> (1)  ]           ]
[           [ NAME    = ‘Kim’ ]           ]
```

Using indices

```
fs = FeatStruct("[\
     NAME=Lee, \
     ADDRESS=(1)[NUMBER=74, STREET=‘rue Pascal’],\
     SPOUSE=[NAME=Kim, ADDRESS->(1)],]")
```

```
[ ADDRESS = (1) [ NUMBER = 74           ] ]
[               [ STREET = ‘rue Pascal’ ] ]
[                                         ]
[ NAME    = ’Lee’                         ]
[                                         ]
[ SPOUSE  = [ ADDRESS -> (1)  ]           ]
[           [ NAME    = ‘Kim’ ]           ]
```

The index is assigned a value, and then referenced.

Using indices: reference before assignment

```
fs = FeatStruct("[\
     NAME=Lee, \
     SPOUSE=[NAME=Kim, ADDRESS->(1)], \
     ADDRESS=(1)[NUMBER=74, STREET=‘rue Pascal’]]")
```

Using indices: reference before assignment

```
fs = FeatStruct("[\
     NAME=Lee, \
     SPOUSE=[NAME=Kim, ADDRESS->(1)], \
     ADDRESS=(1)[NUMBER=74, STREET=‘rue Pascal’]]")

 ValueError: Error parsing feature structure
    [NAME=Lee,
     SPOUSE=[NAME=Kim, ADDRESS->(1)],  ...
                                 ^ Expected bound identifier
```

Using variables

```
from nltk.sem.logic import Variable

fs10=FeatStruct("\
        [NP=[head=[agreement=?n]], \
        VP=[head=(1)[agreement=?n]]]")
```

Using variables

```
from nltk.sem.logic import Variable

fs10=FeatStruct("\
        [NP=[head=[agreement=?n]], \
        VP=[head=(1)[agreement=?n]]]")

[ NP = [ head = [ agreement = ?n ] ] ]
[                                    ]
[ VP = [ head = [ agreement = ?n ] ] ]
```

Example from textbook

What does this feature structure represent?

$$\begin{bmatrix} \text{S} & \begin{bmatrix} \text{head} & \boxed{1} \end{bmatrix} \\ \text{NP} & \begin{bmatrix} \text{head} & \begin{bmatrix} \text{agreement} & \boxed{2} \end{bmatrix} \end{bmatrix} \\ \text{VP} & \begin{bmatrix} \text{head} & \boxed{1} \begin{bmatrix} \text{agreement} & \boxed{2} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Example from textbook

What does this feature structure represent?

$$\begin{bmatrix} \text{S} & \begin{bmatrix} \text{head} & \boxed{1} \end{bmatrix} \\ \text{NP} & \begin{bmatrix} \text{head} & \begin{bmatrix} \text{agreement} & \boxed{2} \end{bmatrix} \end{bmatrix} \\ \text{VP} & \begin{bmatrix} \text{head} & \boxed{1} \begin{bmatrix} \text{agreement} & \boxed{2} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$S \longrightarrow NPVP$

Example from textbook

What does this feature structure represent?

$$\begin{bmatrix} \text{S} & \begin{bmatrix} \text{head} & \boxed{1} \end{bmatrix} \\ \text{NP} & \begin{bmatrix} \text{head} & \begin{bmatrix} \text{agreement} & \boxed{2} \end{bmatrix} \end{bmatrix} \\ \text{VP} & \begin{bmatrix} \text{head} & \boxed{1} \begin{bmatrix} \text{agreement} & \boxed{2} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$S \rightarrow NPVP$

$\langle$ NP head agreement $\rangle = \langle$ VP head agreement $\rangle$

Example from textbook

What does this feature structure represent?

$$\begin{bmatrix} \text{S} & \begin{bmatrix} \text{head} & \boxed{1} \end{bmatrix} \\ \text{NP} & \begin{bmatrix} \text{head} & \begin{bmatrix} \text{agreement} & \boxed{2} \end{bmatrix} \end{bmatrix} \\ \text{VP} & \begin{bmatrix} \text{head} & \boxed{1} \begin{bmatrix} \text{agreement} & \boxed{2} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$S \longrightarrow NPVP$

$\langle$ NP head agreement $\rangle = \langle$ VP head agreement $\rangle$

$\langle$ S head $\rangle = \langle$ VP head $\rangle$

Need to combine variables and indices

```
fs3=FeatStruct("\
       [NP=[head=[agreement=?n]], \
       VP=[head=(1)[agreement=?n]], \
       S=[head->(1)]]")
```

Need to combine variables and indices

```
fs3=FeatStruct("\
       [NP=[head=[agreement=?n]], \
       VP=[head=(1)[agreement=?n]], \
       S=[head->(1)]]")
```

```
[ NP = [ head = [ agreement = ?n ] ]     ]
[                                        ]
[ S  = [ head = (1) [ agreement = ?n ] ] ]
[                                        ]
[ VP = [ head -> (1) ]
```

Unification in NLTK

```
[ VP = [ head = [ agreement = [ number = ‘singular’ ] ] ] ]
[      [        [             [ person = ‘third’    ] ] ] ]

[ NP = [ head = [ agreement = ?n ] ]      ]
[                                         ]
[ S  = [ head = (1) [ agreement = ?n ] ] ]
[                                         ]
[ VP = [ head -> (1) ]                    ]
```

Unification in NLTK

```
[ VP = [ head = [ agreement = [ number = ‘singular’ ] ] ] ]
[      [        [             [ person = ‘third’    ] ] ] ]

[ NP = [ head = [ agreement = ?n ] ]     ]
[                                        ]
[ S  = [ head = (1) [ agreement = ?n ] ] ]
[                                        ]
[ VP = [ head -> (1) ]                   ]

[ NP = [ head = [ agreement = (1) [ number = ‘singular’ ] ] ] ]
[      [        [                 [ person = ‘third’    ] ] ] ]
[                                                             ]
[ S  = [ head = (2) [ agreement -> (1) ] ]                    ]
[                                                             ]
[ VP = [ head -> (2) ]                                        ]
```

Printed version looks odd, but graph is intact.