Feature-based Grammar

Ling 571
Deep Techniques for NLP
February 2, 2001
Roadmap

- Implementing feature-based grammars
  - Features in NLTK
- Designing feature grammars
  - A Complex Agreement Example
- Semantic features
Summary

- Features defined
- Modeling features:
  - Attribute-Value Matrices (AVM)
  - Directed Acyclic Graph (DAG)
- Mechanisms for features:
  - Subsumption
  - Unification
- Parsing with features:
  - Augmenting the Earley parser
Feature Grammar in NLTK

- NLTK supports feature-based grammars
  - Includes ways of associating features with CFG rules

- Includes readers for feature grammars
  - .fcfg files

- Includes parsers
  - Nltk.parse.FeatureEarleyChartParse
Creating and Accessing NLTK Feature Structures

- Create with *FeatStruct*
Creating and Accessing NLTK Feature Structures

- Create with `FeatStruct`

```python
>>> fs1 = nltk.FeatStruct(NUMBER='pl', PERSON=3)
>>> print fs1
[ NUMBER = 'pl'
[ PERSON = 3 ]

>>> print fs1['NUMBER']
pl

>>> fs1['NUMBER'] = 'sg'
```
Creating and Accessing NLTK Feature Structures

- Create with *FeatStruct*

  ```python
  >>> fs1 = nltk.FeatStruct(NUMBER='pl', PERSON=3)
  ```
Creating and Accessing NLTK Feature Structures

- Create with *FeatStruct*

  ```python
  >>> fs1 = nltk.FeatStruct(NUMBER='pl', PERSON=3)
  >>> print fs1
  [ NUMBER = 'pl’ ]
  [ PERSON = 3 ]
  ```
Creating and Accessing NLTK Feature Structures

- Create with `FeatStruct`

  ```python
  >>> fs1 = nltk.FeatStruct(NUMBER='pl', PERSON=3)
  >>> print fs1
  [ NUMBER = 'pl' ]
  [ PERSON = 3 ]
  >>> print fs1['NUMBER']
  pl
  ```
Creating and Accessing NLTK Feature Structures

- Create with `FeatStruct`

```python
>>> fs1 = nltk.FeatStruct(NUMBER='pl', PERSON=3)
>>> print fs1
[ NUMBER = 'pl'
 [ PERSON = 3 ]

>>> print fs1['NUMBER']
pl
>>> fs1['NUMBER'] = 'sg'
```
Complex Feature Structures

- >>> fs2 = nltk.FeatStruct(POS='N', AGR=fs1)
Complex Feature Structures

- >>> fs2 = nltk.FeatStruct(POS='N', AGR=fs1)
- >>> print fs2
- [ POS = 'N'    ]
- [ [ NUMBER = 'sg' ] ]
- [ AGR = [ PERSON = 3 ] ]
Complex Feature Structures

- >>> fs2 = nltk.FeatStruct(POS='N', AGR=fs1)
- >>> print fs2
- [ POS = 'N'  ]
- [            [ NUMBER = 'sg' ]  ]
- [ AGR = [ PERSON = 3 ] ]

- Alternatively,
- >>> fs3 = nltk.FeatStruct("[POS='N',
  AGR=[NUM='pI',PER=3]]")


Reentrant Feature Structures

- First instance
  - Parenthesized integer: (1)
Reentrant Feature Structures

- First instance
  - Parenthesized integer: (1)

- Subsequent instances:
  - ‘Pointer’: -> (1)
Reentrant Feature Structures

- First instance
  - Parenthesized integer: (1)

- Subsequent instances:
  - ‘Pointer’: -> (1)

- >>> print nltk.FeatStruct("[A='a', B=(1)[C='c'], D->(1)]")
Reentrant Feature Structures

- First instance
  - Parenthesized integer: (1)

- Subsequent instances:
  - ‘Pointer’: -> (1)

- `>>> print nltk.FeatStruct("[A='a', B=(1)[C='c'], D->(1)]")`
  - [ A = ‘a’ ]
  - [ B = (1) [ C = ‘c’ ]]
  - [ D -> (1) ]
Augmenting Grammars

- Attach feature information to non-terminals, on
  
  - N[AGR=[NUM='pl']] -> 'students'
  - N[AGR=[NUM='sg']] -> 'student'
Augmenting Grammars

- Attach feature information to non-terminals, on:
  - $N[\text{AGR}=[\text{NUM}='\text{pl}']]] \rightarrow '\text{students}'$
  - $N[\text{AGR}=[\text{NUM}='\text{sg}']]] \rightarrow '\text{student}'$

- So far, all values are literal or reentrant
Augmenting Grammars

- Attach feature information to non-terminals, on
  - N[AGR=[NUM='pl']] -> 'students'
  - N[AGR=[NUM='sg']] -> 'student'

- So far, all values are literal or reentrant
  - Variables allow generalization: ?a
    - Allows underspecification, e.g. Det[GEN=?a]
Augmenting Grammars

- Attach feature information to non-terminals, on
  - N[AGR=[NUM='pl']] -> 'students'
  - N[AGR=[NUM='sg']] -> 'student'

- So far, all values are literal or reentrant
  - Variables allow generalization: ?a
    - Allows underspecification, e.g. Det[GEN=?a]
Mechanics

- >>> fs3 = nltk.FeatStruct(NUM='pl', PER=3)

- >>> fs4 = nltk.FeatStruct(NUM='pl')

- >>> print fs4.unify(fs3)

- [NUM = 'pl']
- [PER = 3]
Morphosyntactic Features

- Grammatical feature that influences morphological or syntactic behavior

- English:
  - Number:
    - Dog, dogs
  - Person:
    - Am; are; is
  - Case:
    - I – me; he – him; etc
  - Countability:
More Complex German Example

- Subject – singular, masc
  - *der Hund*
  - The dog

- Subject – plural, masc
  - *die Hunde*
  - The dogs
More Complex German Example

- Objects – determined by verb
- Dative – singular, masc
  - *dem Hund*
  - The dog
- Accusative – plural, masc
  - *die Hunde*
  - The dogs
Contrast

- Subject:
  - *Die Katze*
  - The cat

- Subject: plural
  - *Die Katze*
  - The cats
Contrast

- Object:
  - *Die Katze*
  - The cat

- Object:
  - *Der Katze*
  - The cat
Analysis

- What are the key contrasts?
  - Number
    - Singular, plural
  - Gender
    - Masc, Fem, ....
  - Case:
    - Subject (nom), dative, accusative, ....

+ Interactions
Feature Interaction

- Interactions of German case, number, gender

<table>
<thead>
<tr>
<th>Case</th>
<th>Masc</th>
<th>Fem</th>
<th>Neut</th>
<th>PL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nom</td>
<td>Der</td>
<td>Die</td>
<td>Das</td>
<td>Die</td>
</tr>
<tr>
<td>Gen</td>
<td>Des</td>
<td>Der</td>
<td>Des</td>
<td>Den</td>
</tr>
<tr>
<td>Dat</td>
<td>Dem</td>
<td>Der</td>
<td>Dem</td>
<td>Den</td>
</tr>
<tr>
<td>Acc</td>
<td>Den</td>
<td>Die</td>
<td>Das</td>
<td>Die</td>
</tr>
</tbody>
</table>
Examples of Interaction

<table>
<thead>
<tr>
<th>Die</th>
<th>Katze</th>
<th>Sieht</th>
<th>Den</th>
<th>Hund</th>
</tr>
</thead>
<tbody>
<tr>
<td>The cat sees the dog</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Examples of Interaction

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The cat sees the dog</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The cat sees the dog</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Examples of Interaction

<table>
<thead>
<tr>
<th>English</th>
<th>German</th>
<th>Verb</th>
<th>Case</th>
<th>German</th>
<th>German</th>
</tr>
</thead>
<tbody>
<tr>
<td>The cat sees the dog</td>
<td>Die Katze Sieht Den Hund</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The cat helps the dog</td>
<td>Die Katze hilft Dem Hund</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Note: The examples marked with an asterisk (*) indicate an alternative word order or usage.*
### Examples of Interaction

<table>
<thead>
<tr>
<th>German Verbs</th>
<th>English Translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Die The Nom. Fem. sg</td>
<td>The cat sees the dog</td>
</tr>
<tr>
<td>Katze Cat. 3. Fem. sg</td>
<td>Sieht See. 3. sg</td>
</tr>
<tr>
<td>Den The Acc. Masc. sg</td>
<td>Hund Dog. 3. Masc. sg</td>
</tr>
<tr>
<td>Die The Nom. Fem. sg</td>
<td>The cat helps the dog</td>
</tr>
<tr>
<td>Katze Cat. 3. Fem. sg</td>
<td>hilft help. 3. sg</td>
</tr>
<tr>
<td>Dem The Dat. Masc. sg</td>
<td>Hund Dog. 3. Masc. sg</td>
</tr>
</tbody>
</table>

German verbs in, at least, 2 classes: assign diff’t object case
Semantic Features

- Grammatical features that influence semantic (meaning) behavior of associated units

- E.g.:
Semantic Features

- Grammatical features that influence semantic (meaning) behavior of associated units

- E.g.:
  - ?The rocks slept.
Semantic Features

- Grammatical features that influence semantic (meaning) behavior of associated units

- E.g.:
  - ?The rocks slept.
  - ?Colorless green ideas sleep furiously.
Semantic Features

- Many proposed:
  - Animacy: +/-
  - Natural gender: masculine, feminine, neuter
  - Human: +/-
  - Adult: +/-
  - Liquid: +/-
  - Etc.
  - The milk spilled.
  - ?The cat spilled.
Examples

- The climber hiked for six hours.
Examples

- The climber hiked for six hours.
- The climber hiked on Saturday.
Examples

- The climber hiked for six hours.
- The climber hiked on Saturday.
- The climber reached the summit on Saturday.
Examples

- The climber hiked for six hours.
- The climber hiked on Saturday.
- The climber reached the summit on Saturday.
- *The climber reached the summit for six hours.

Contrast:
Examples

- The climber hiked for six hours.
- The climber hiked on Saturday.
- The climber reached the summit on Saturday.
- *The climber reached the summit for six hours.

Contrast:
- Achievement vs activity
Semantic features & Parsing

- Can filter some classes of ambiguity

- Old men and women slept.
Semantic features & Parsing

- Can filter some classes of ambiguity
  - Old men and women slept.
  - (Old men) and (women) slept.
Semantic features & Parsing

- Can filter some classes of ambiguity
  - Old men and women slept.
  - (Old men) and (women) slept.
  - (Old (men and women)) slept.
Semantic features & Parsing

- Can filter some classes of ambiguity

- Old men and women slept.
- (Old men) and (women) slept.
- (Old (men and women)) slept.

- Sleeping people and books lie flat.
Semantic features & Parsing

- Can filter some classes of ambiguity
  - Old men and women slept.
  - (Old men) and (women) slept.
  - (Old (men and women)) slept.

- Sleeping people and books lie flat.
  - (Sleeping people) and (books) lie flat.
Semantic features & Parsing

- Can filter some classes of ambiguity
  - Old men and women slept.
  - (Old men) and (women) slept.
  - (Old (men and women)) slept.
  - Sleeping people and books lie flat.
  - (Sleeping people) and (books) lie flat.
  - (Sleeping (people and books)) lie flat.
Semantic features & Parsing

- Can filter some classes of ambiguity

- Old men and women slept.
- (Old men) and (women) slept.
- (Old (men and women)) slept.

- Sleeping people and books lie flat.
- (Sleeping people) and (books) lie flat.
- *(Sleeping (people and books ))lie flat.
Summary

- **Features**
  - Enable compact representation of grammatical constraints
  - Capture basic linguistic patterns

- **Unification**
  - Creates and maintains consistency over features

- Integration with parsing allows filtering of ill-formed analyses