Unification Grammar

Problems in "symbol-based" grammars

* the ducks flies * the swallow flies ducks

lexicon:
<table>
<thead>
<tr>
<th>the</th>
<th>Det (definite)</th>
</tr>
</thead>
<tbody>
<tr>
<td>little</td>
<td>Adj (size)</td>
</tr>
<tr>
<td>orange</td>
<td>Adj (colour)</td>
</tr>
<tr>
<td>Noun (fruit)</td>
<td></td>
</tr>
<tr>
<td>ducks</td>
<td>Noun (animal)</td>
</tr>
<tr>
<td>Verb (action)</td>
<td></td>
</tr>
<tr>
<td>swallow</td>
<td>Noun (animal)</td>
</tr>
<tr>
<td>Verb (action)</td>
<td></td>
</tr>
<tr>
<td>flies</td>
<td>Noun (animal)</td>
</tr>
<tr>
<td>Verb (action)</td>
<td></td>
</tr>
</tbody>
</table>

Lexicon:
* the ducks flies, * the swallow flies ducks

ducks:: duck | ducts
drink:: drink | drank | drunk | drinks

Agreement

- http://www.w3.org/TR/speech-grammar/#S6.2 (aug. 01)
  - "The morphological variations of a token may depend on the grammatical class of the
token (e.g., verb, noun, male noun, adjective). Morphological rules are typically
language-specific and some languages have much richer morphological behaviour than
others." [...] Example (pseudo format)
  - dog:: dog | dogs
  - drink:: drink | drank | drunk | drinks

- Status: The Working Group is not aware of any existing grammar format that
supports this kind of morphological inference by a speech recognizer for a grammar.
- The grammar format "supporting" this "inference" problem is unification grammar!!!
UNIFICATION GRAMMAR

- Actual origin in unpublished documents by M. Kay in the seventies
- Classical presentations:

PSG/BNF
- symbols
- parsing: symbols are string-compared
- comparison of two symbols returns TRUE or FALSE
- the set of symbols is static

UNIFICATION GR/APSG
- feature sets
- parsing: feature sets are unified
- unification of two feature sets returns a new feature set or NULL
- the set of feature sets is dynamic

![Feature set diagram]

{cat=s, stype=declarative}

Feature set

attribute

value

feature
UNIFICATION GRAMMAR

Terminology:
- attribute: lex, cat
- value: s, np, vp, fly ...
- feature: lex=fly ...
- atomic feature: lex=fly
- complex feature*: agreement={prs=third, nb=plur}
- constant feature: prs=third
- variable feature**: prs=$A

* NOT supported by all unification grammars (e.g. by the CPK NLP Suite)
** I literature, often indicated by numbered boxes prs=

Unification: the basic operation on compound features

{cat=n} {lex=flies, cat=n} = {lex=flies, cat=n}
{cat=v} {lex=flies, cat=n} = NULL
{cat=pron, nb=plur} {lex=you, cat=pron} = {lex=you, cat=pron, nb=plur}
{cat=pron, nb=sing} {lex=you, cat=pron} = {lex=you, cat=pron, nb=sing}
{cat=pron, nb=plur} {lex=I, cat=pron, nb=sing} = NULL

Unification and variables 1):

SN is instantiated:
{cat=pron, nb=$N} {lex=I, cat=pron, nb=sing} = {lex=I, cat=pron, nb=sing}

SN remains uninstantiated:
{cat=pron, nb=$N} {lex=you, cat=pron} = {lex=you, cat=pron, nb=$N}
UNIFICATION GRAMMAR

Unification and variables 2):
Scope of SN is the entire structure building rule ("shared structures")
(dotted rule chart parsing scheme indicated by ?x)

\[ \{ \text{cat} = \text{np} , \text{nb} = \text{SN} \} \]
\[ \{ \text{cat} = \text{pron} , \text{nb} = \text{SN} \} \]
\[ \{ \text{lex} = \text{I} , \text{cat} = \text{pron} , \text{nb} = \text{sing} \} \]

\[ \{ \text{cat} = \text{np} , \text{nb} = \text{sing} \} \]

\[ \{ \text{lex} = \text{I} , \text{cat} = \text{pron} , \text{nb} = \text{sing} \} \]

Unification and variables 3):
Scope of SN is the entire structure building rule
(dotted rule chart parsing scheme indicated by ?x)

\[ \{ \text{cat} = \text{s} \} \]
\[ \{ \text{cat} = \text{np} , \text{nb} = \text{SN} \} \]
\[ \{ \text{cat} = \text{vp} , \text{nb} = \text{SN} \} \]
\[ \{ \text{cat} = \text{np} , \text{nb} = \text{sing} \} \]
\[ \{ \text{lex} = \text{I} , \text{cat} = \text{pron} , \text{nb} = \text{sing} \} \]

\[ \{ \text{cat} = \text{np} , \text{nb} = \text{sing} \} \]
\[ \{ \text{cat} = \text{vp} , \text{nb} = \text{sing} \} \]

UNIFICATION GRAMMAR

Unification and variables 4):
Agreement a variable occurs twice or more in the body of a rule:

\[ \{ \text{cat} = \text{styp = decl} \} \]
\[ \{ \text{cat} = \text{np} , \text{prs} = \text{SN} \} \]
\[ \{ \text{cat} = \text{vp} , \text{prs} = \text{SN} \} \]

i.e.: subject - predicate agreement in person and number:
I am, you are, the man is, the men are, *I am, *you are...
UNIFICATION GRAMMAR

Unification and variables 5):

Percolation: A variable occurs once in the head and once (or more) in the body of a rule:

\[ \{ \text{cat=np, prs=third, nb=$N,} \} \]
\[ \{ \text{cat=det, } \} \quad \text{optional determiner "a", "the"} \}
\[ \{ \text{cat=adj, } \} \quad \text{zero or more adjectives} \}
\[ \{ \text{cat=n, nb=$N} \} \quad \text{obligatory noun} \}

i.e.: the number of an NP(subject,object) is the number of the noun contained in the NP, the man (sing), the large tall man (sing), men (plur.), the men (plur.)

UNIFICATION GRAMMAR

The special "cat"-feature, obligatory in most formalisms:

• historical reasons: In the earliest approaches, features were introduced as additional "feature constraints" to the PSG:

\[ s \rightarrow \text{np vp} \]
\[ s.\text{stype}=\text{decl} \]
\[ \text{np.prs=vp.prs} \]
\[ \text{np.nb=vp.nb} \]
\[ \{ \text{cat=s, stype= decl} \} \]
\[ \{ \text{cat= np ,prs=$P, nb=$N} \} \]
\[ \{ \text{cat= vp, prs=$P,nb =$N} \} \]

• Parsing: Many efficient parsing algorithm developed for normal context-free grammars (like Earley) can only be applied on feature-based grammars with at least one obligatory feature

• Prevents unification from "exploding".

UNIFICATION GRAMMAR

Another special feature: "lex"

• In some formalisms, lexical rules consists of identifier+feature set:

  "man": \{ \text{cat=noun, number=sing, case=non_genitive} \}

• In other formalisms, the identifier is a feature, bound to the attribute "lex", that can participate in unification the normal way:

  \{ \text{lex=man,cat=noun, number=sing, case=non_genitive} \}
UNIFICATION GRAMMAR

Shieber 1986:

- General demands on grammar formalisms
- Linguistic felicity
- Expressiveness
- Computational effectiveness

Characteristics of Unification Grammars

- Surface-based
- Informational
- Inductive
- Declarative

CPK NLP Suite

- aps, psg, icm, voc
- pars, apspars, psgpars
- trec, apstrec
- tslu, apstslu
- conv

VoiceXML 1.0
### W3C Voice Browser Activity

Status Feb 2002 (http://www.w3.org/Voice)

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speech grammars</td>
<td>Speech Recognition Grammar -&gt; JSGF</td>
</tr>
<tr>
<td></td>
<td>Stochastic Language Models</td>
</tr>
<tr>
<td>Voice dialogs</td>
<td>Semantic Interpretation Markup Language</td>
</tr>
<tr>
<td></td>
<td>NEW: Dialog Markup Language: VoiceXML 2.0 = &quot;Memorandum of Understanding&quot;</td>
</tr>
<tr>
<td>Speech synthesis</td>
<td>Speech Synthesis Markup Language</td>
</tr>
<tr>
<td></td>
<td>NEW: Speech Markup Language</td>
</tr>
<tr>
<td>Natural language representation</td>
<td>Natural Language Semantics Markup Language</td>
</tr>
<tr>
<td>Multimodal systems</td>
<td>NEW: given up Multimodal Dialog Markup Language</td>
</tr>
<tr>
<td>Reusable dialog components</td>
<td>NEW: given up Reusable Dialog</td>
</tr>
</tbody>
</table>

### VoiceXML “Hello World”

```xml
<?xml version="1.0"?>
<vxml version="1.0">
<form>
<field name="drink">
<prompt>Would you like coffee, tea, milk, or nothing?</prompt>
<grammar src="drink.gram" type="application/x-jsgf"/>
</field>
<block>
<submit next="http://www.drink.example/drink2.asp"/>
</block>
</form>
</vxml>
```

### VoiceXML Overview
VoiceXML has good dialog components

- **Menu dialog**
  - Used when a single value/turn leads to the next action
  - Good abstraction for menus and list selection

- **Form dialog**
  - Used when multiple fields must be filled before the next action
  - Good abstraction for information gathering for a transaction

VoiceXML integrates with the Internet

- Designed for a client/server environment
- Transactions are handled by an application server using existing web protocols (HTTP, JSP, CGI, ASP, JavaServlets, …)
- VoiceXML documents, grammars, and sound files are referenced by URLs
- Will probably appeal to web programmers

VoiceXML makes highly dynamic dialogues possible

- VoiceXML documents can be generated “on the fly” by the server
- Form interpretation allows multiple paths through the dialogue
- Embedded ECMAScript allows “wild” constructions
  - expression can calculate the next state
  - field variables can be changed
Apparent simplicity hides the complexity of dialogue design

- Making a spoken dialogue application is still a complex task
- Speech recognition performance is still a limiting factor
- A deeper understanding of the underlying technologies is still required

VoiceXML applications may have limited portability

- Differences in the implementation of VoiceXML portals is likely to impact the performance of VoiceXML applications.
- Speech recognition performance will vary between portals
- Differences in speech synthesis will influence prompting, which may in turn influence user responses

Verification of dialogue applications can be difficult

- The form filling may lead to multiple dialogue paths
- The use of embedded ECMA Script in a dialogue increases the risk of bugs surfacing as runtime errors
- Dialogues may execute remotely in an environment which is outside the control of the dialogue programmer
VoiceXML is not suitable for embedded dialogues

- Designed for a client/server environment
- Needs a large support structure (a browser)
  - Generic speech recognition:
    - Dictation mode, JSGF, word spotting
  - Text to Speech Synthesis
- Verification is difficult/costly
  - Form filling leads to multiple dialogue paths
  - ECMAScript interpretation

VoiceXML summary

- Sets a standard for voice portals - supported by W3C (new!)
- We should not expect it to improve the performance of spoken dialogue applications
- We may see a proliferation of poorly performing spoken dialogue applications